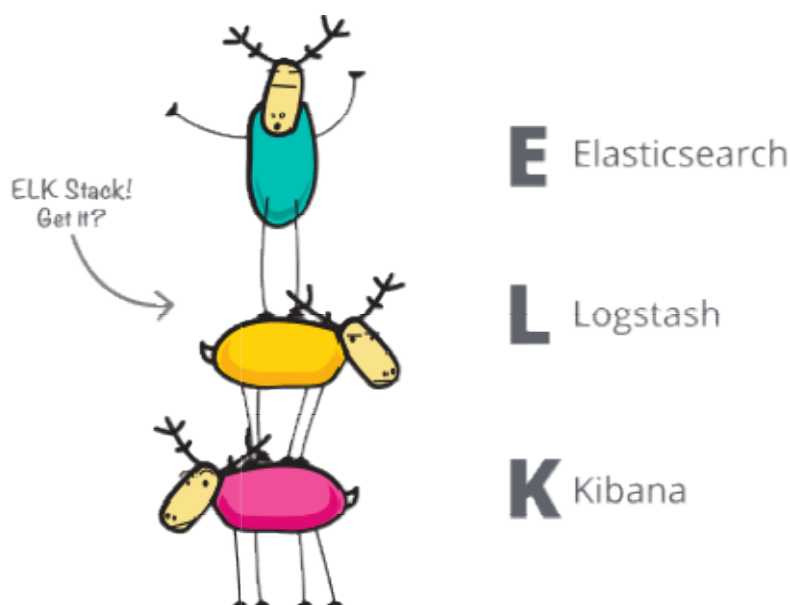


Introduction au cours ELK

Introduction au cours complet pour débutants afin d'apprendre à utiliser les différents composants de la suite ELK.

Introduction

Hello les champions, ça vous tente de superviser vos logs de manières différentes ? Je suppose que la réponse est oui ☐! Dans ce guide, nous allons découvrir **pourquoi l'analyse de log connaît-il un tel intérêt** et nous allons jeter un regard complet sur les différentes technologies composant la pile ? Le but est de comprendre quel rôle ils jouent dans vos analyses de données. Enfin dans les futurs chapitres de ce cours nous verrons plus en détail comment les installer, les configurer et les utiliser et comment éviter au mieux certains pièges courants en cours de route.



Public visé

Cette série d'articles est conçue pour les débutants ayant besoin de comprendre l'utilisation de la suite ELK à partir de zéro. Ce tutoriel vous donnera une compréhension suffisante de la technologie, qui vous permettra plus tard d'atteindre des niveaux d'expertise beaucoup plus élevés.

Prérequis

Ce cours ne demande pas forcément de prérequis que vous devriez au minimum avoir. Cependant si vous maîtrisez déjà ou que vous avez une compréhension générale d'une solution de supervision et que vous êtes à l'aise avec l'environnement Linux, alors il vous sera très facile de comprendre les différents concepts de la suite ELK et d'avancer rapidement sur votre piste d'apprentissage, mais ce n'est pas non plus indispensable !

Sans plus attendre Let's go !

L'analyse de logs

Les logs sont l'une des informations les plus précieuses en matière de gestion et de surveillance des systèmes informatiques. Comme ils enregistrent toutes les actions qui ont eu lieu sur une machine, ils fournissent des informations dont vous avez besoin pour repérer les problèmes susceptibles d'avoir un impact sur les **performances**, la **conformité** et la **sécurité** des éléments de votre infrastructure. C'est pourquoi la gestion des logs devrait faire partie de toute infrastructure de surveillance.

Lorsqu'on parle d'analyse de logs, le plus important défi consiste à regrouper, normaliser, visualiser et analyser vos logs dans un emplacement unique et accessible, d'où l'utilisation de la suite ELK.

Qu'est-ce que l'analyse des logs?

L'analyse de logs est le processus consistant à donner du sens aux messages de logs générés par votre système dans le but d'utiliser ces données pour améliorer ou résoudre des problèmes de performances au sein d'une application ou d'une infrastructure. Dans une vue d'ensemble, les entreprises analysent les logs pour atténuer les risques de manière proactive et réactive, se conformer aux politiques de sécurité, aux audits et aux réglementations, et permet également de mieux comprendre le comportement de vos utilisateurs utilisant vos applications.

Pourquoi l'analyse des logs est-elle importante?

La plupart des entreprises sont tenues responsable d'effectuer l'archivage et l'analyse des logs dans le cadre de leurs réglementations de conformité. Ils doivent régulièrement surveiller et analyser les logs des systèmes pour rechercher des erreurs, des anomalies ou des activités

suspectes ou non autorisées qui s'écartent de la norme. Cela leur permet de recréer la chaîne d'événements qui a conduit à un problème et de le résoudre efficacement.

Par exemple, vous pourrez rechercher des erreurs HTTP et comprendre où et pourquoi elles se sont produites ou détecter si les utilisateurs ne reçoivent pas les informations recherchées ou si le chargement de leurs demandes prend trop de temps, ou si certains microservices rencontrent des problèmes, etc ...

Lorsque vous tirez parti de l'analyse des logs, vous pouvez détecter les problèmes avant ou lorsqu'ils surviennent et éviter le gaspillage de temps, les retards inutiles et les coûts supplémentaires qui vont avec. Ainsi, les équipes peuvent intervenir et résoudre les problèmes plus rapidement, leurs permettant ainsi de se concentrer davantage sur l'amélioration des fonctionnalités existantes et sur l'ajout de nouvelles fonctionnalités aux produits et services qu'ils créent au lieu de passer du temps à dépanner manuellement. Cela, à son tour, augmente la valeur des logiciels qu'ils construisent et conduit à des versions plus fréquentes (DevOps) et augmente la valeur globale pour l'entreprise.

Meilleures pratiques d'analyse des logs

L'analyse des logs est un processus complexe qui doit suivre les bonnes pratiques suivantes :

- **La détection et la reconnaissance des patterns** : cela fait référence au filtrage des messages entrants sur la base d'un pattern souvent combiné avec l'utilisation des expressions régulières. Elle fait partie intégrante de l'analyse des logs car elle permet de détecter plus rapidement les anomalies.
- **La normalisation des logs** : ce sont le processus de conversion des éléments de logs tels que les adresses IP ou les horodatages, dans un format commun pour toutes vos équipes.
- **La classification et le balisage** : ce sont le processus de balisage (tag) des messages avec des mots-clés et de leur catégorisation en classes. Cela vous permet de filtrer et de personnaliser la façon dont vous visualisez les données.
- **L'analyse de corrélation** : fait référence à la collecte de données provenant de différentes sources et à la recherche de messages appartenant à un événement spécifique. Par exemple, en cas d'activité malveillante, il vous permet de filtrer et de corréler les logs provenant de vos périphériques réseaux, pare-feu, serveurs et autres sources afin de très rapidement détecter la source du problème.
- **Alerte** : L'analyse de corrélation est généralement associée aux systèmes d'alerte, en fonction du pattern que vous avez identifié, vous pouvez créer des alertes lorsque votre analyseur de logs détecte une activité anormale.

ELK

C'est quoi ELK ?

La stack ELK est un acronyme utilisé pour décrire une pile qui comprend trois projets open sources populaires : Elasticsearch, Logstash et Kibana. Elle est la principale **solution open source de gestion des logs pour les entreprises** qui souhaitent bénéficier des avantages d'une **solution de journalisation centralisée**.

En effet, les outils : Elasticsearch, Logstash et Kibana, lorsqu'ils sont utilisés ensemble (car oui on peut les utiliser séparément ou avec d'autres technologies), forment une pile de bout en bout offrant une analyse de données dont des logs en temps réel afin de fournir des informations exploitables à partir de presque tout type de source de données structurée et non structurée. Vous comprendrez plus tard que chacun de ces produits joue un rôle différent dans vos analyses. Ils peuvent être utilisés pour des projets simples ou complexe car ils prennent en charge des opérations simples et avancées.

Les rôles des composants

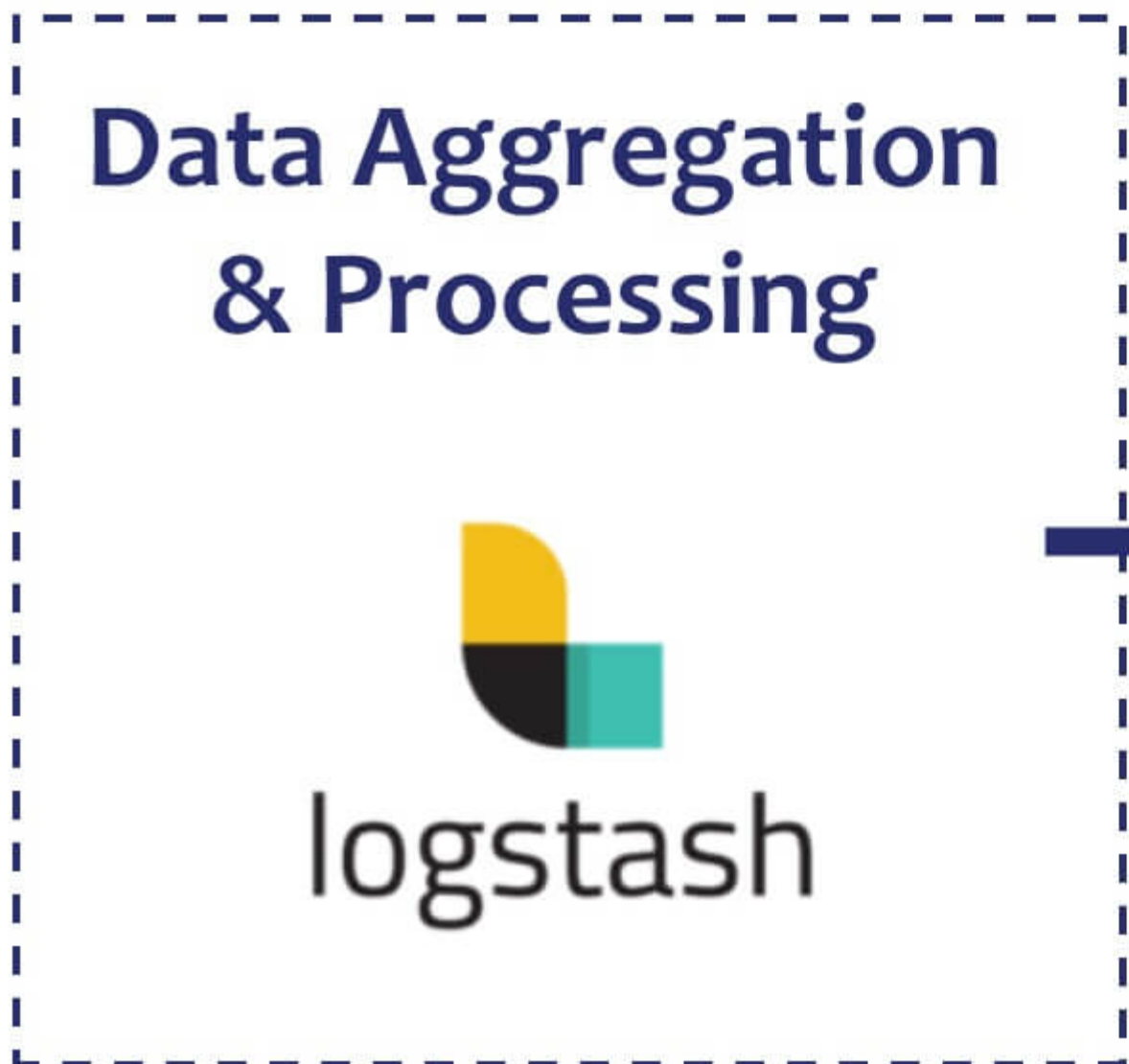
Il est important de dissocier le rôle de ces 3 outils afin de mieux comprendre le rôle et la communication de l'ensemble de ces composants :

- **Elasticsearch** : en charge de l'indexation et du stockage de vos informations sur une base de données NoSQL qui est basé sur le moteur de recherche **Apache Lucene** et il est construit pour fournir des APIS rest. Il offre un déploiement simple, une fiabilité maximale et une gestion facile. Il propose également des requêtes avancées pour effectuer une analyse détaillée et stocke toutes les données de manière centralisée. Il est également utilisé sur de nombreux projets hors la suite ELK car il permet d'exécuter une recherche rapide des documents.
- **Logstash** : outil d'intégration de données open source qui vous permet de collecter des données à partir d'une variété de sources, de les filtrer, les transformer et de les envoyer à la destination souhaitée (ex: Elasticsearch). Son but principal est de rassembler et normaliser tous les types de données provenant de différentes sources et de les rendre disponibles pour une utilisation ultérieure.
- **Kibana** : outil de visualisation de données qui complète la pile ELK , c'est une couche de visualisation qui fonctionne au-dessus d'Elasticsearch, offrant aux utilisateurs la possibilité d'analyser et de visualiser les données récupérées Elasticsearch. C'est un outil puissant offrant pour vos tableaux de bord divers diagrammes interactifs, données géographiques et graphiques pour visualiser les données les plus complexes.

Récemment, un nouvel outil a été rajouté à la Stack Elastic nommé **Beats** il est très utile si vous utilisez plusieurs machines car ça reste un agent léger et réservé au transfert de données, il se charge de transférer les données vers Logstash et Elasticsearch. Nous l'utiliserons dans un futur chapitre.

Comment fonctionnent-ils ensemble ?

Comme il l'est mentionné ci-dessus, prit ensemble, les différents composants de la pile ELK fournissent une solution simple mais puissante pour la gestion et l'analyse des logs.



Logstash sera utilisé pour récupérer, filter et normaliser des informations liées à vos logs issues de différents sources. Une fois cela fait, Logstash envoie ces informations dans un système de stockage ici Elasticsearch qui quant à lui se chargera d'indexer, stocker et

effectuer une recherche et une analyse en temps réel de vos données. Enfin, Kibana récupère ces données afin de fournir un système de visualisation et d'exploration en plus de Logstash et Elasticsearch afin que vous puissiez facilement comprendre vos données sous forme de tableaux et de graphiques.

Information

Cependant, pour gérer des pipelines plus complexes conçus pour gérer de grandes quantités de données en production, des composants supplémentaires sont susceptibles d'être ajoutés à votre architecture de journalisation comme par ex : Kafka, RabbitMQ, Redis.

Études de cas

Voici quelques entreprises populaires qui utilisent la suite ELK :

- Netflix s'appuie fortement sur la pile ELK. L'entreprise utilise la pile ELK pour surveiller et analyser les logs de sécurité des opérations du service client. Il leur permet d'indexer, de stocker et de rechercher des documents à partir de plus de quinze clusters qui comprennent près de 800 nœuds.
- LinkedIn utilise la pile ELK pour surveiller leurs performances et leur sécurité. L'équipe informatique a intégré ELK à Kafka pour comprendre leur charge en temps réel. Leur opération ELK comprend plus de 100 clusters dans six datacenters différents.
- Medium est une célèbre plateforme de publication de blogs. Ils utilisent la pile ELK pour déboguer leurs problèmes de production. De plus, en utilisant cette pile, la société peut prendre en charge 25 millions de lecteurs uniques ainsi que des milliers de publications publiées chaque semaine.

Conclusion

Retrouvons-nous dans le prochain chapitre afin d'installer et configurer notre environnement ELK.

Installation et configuration de la stack ELK

Dans ce chapitre, nous allons procéder à l'installation et la configuration des différents composants de la suite ELK.

Introduction

La stack ELK peut être **installée à l'aide d'une variété de méthodes** et sur un **large éventail de systèmes d'exploitation** et d'**environnements différents**. Vous pouvez **installer ELK localement**, sur le cloud, à l'aide de Docker et de systèmes de gestion de configuration comme Ansible, Puppet et Chef. La pile peut être également installée à l'aide de votre gestionnaire de paquets ou manuellement depuis les binaires officiels. Voici le lien pour la [page officielle des multiples méthodes d'installation d'ELK](#).

De nombreuses étapes d'installation sont similaires d'un environnement à l'autre et comme nous ne pouvons pas couvrir tous les différents scénarios, je vais vous fournir un exemple d'**installation de tous les composants de la pile Elasticsearch, Logstash, Kibana sous une seule machine Linux** à l'aide du gestionnaire de paquets APT et UUM afin de posséder et de gérer les dernières versions des composants.

Information

Pour ceux travaillant sur une machine Windows, veuillez créer une machine virtuelle afin de poursuivre du mieux ce cours.

À savoir

Lors de l'installation d'ELK, vous devez utiliser la même version sur l'ensemble de la pile. Par exemple, si vous utilisez Elasticsearch 7.8.0 alors Kibana doit être aussi sous sa version 7.8.0 et même pour pour Logstash en version 7.8.0.

Si vous mettez à niveau une installation existante, consultez la [page officielle de Mise à niveau de la pile ELK](#).

Ordre d'installation

Pour installer les produits ELK il est recommandé de respecter l'ordre suivant afin de garantir que les composants dont chaque produit dépend sont résolus :

1. Elasticsearch
2. Kibana
3. Logstash

Installation

Elasticsearch

Installation

Tout d'abord, vous devez ajouter la clé de signature d'Elastic pour que le package téléchargé puisse être vérifié (ignorez cette étape si vous avez déjà installé des packages d'Elastic):

Sous la famille debian :

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo  
apt-key add -Copier
```

Sous la famille redhat :

```
rpm --import https://artifacts.elastic.co/GPG-KEY-  
elasticsearchCopier
```

Si vous êtes sur une machine de la famille de Debian, vous devez installer le paquet **apt-transport-https** :

```
sudo apt-get install apt-transport-httpsCopier
```

L'étape suivante consiste à ajouter le dépôt Elasticsearch sur votre système :

Sous la famille debian :

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main"  
| sudo tee -a /etc/apt/sources.list.d/elastic-7.x.listCopier
```


Sous la famille redhat, créez un fichier et nommé le par exemple `elasticsearch.repo` dans le répertoire `/etc/yum.repos.d/`, contenant:

```
[elasticsearch]

name=Elasticsearch repository for 7.x packages

baseurl=https://artifacts.elastic.co/packages/7.x/yum

gpgcheck=1

gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch

enabled=0

autorefresh=1

type=rpm-mdCopier
```

Il ne vous reste plus qu'à mettre à jour vos référentiels et installer Elasticsearch:

Sous la famille debian :

```
sudo apt-get update -y && sudo apt-get install elasticsearchCopier
```

Sous la famille redhat :

```
sudo yum install --enablerepo=elasticsearch elasticsearchCopier
```

Configuration

Par défaut à son lancement Elasticsearch consomme 1go de mémoire de la JVM (machine virtuelle java), si votre machine n'est pas assez puissante vous pouvez modifier les valeurs `Xms` et `Xmx` situé dans le fichier `/etc/elasticsearch/jvm.options` pour une consommation réduite:

```
# Avant (1go)

-Xms1g

-Xmx1g
```

```
# Après (512 mo)

-Xms512mo

-Xmx512moCopier
```

Les configurations Elasticsearch sont effectuées à l'aide du fichier de configuration `/etc/elasticsearch/elasticsearch.yml` qui vous permet de configurer les paramètres généraux comme par exemple le nom du nœud, ainsi que les paramètres réseau comme par exemple l'hôte et le port, l'emplacement des données stockées, la mémoire, les fichiers de logs, etc... Pour ce cours nous laisserons la configuration par défaut.

Lancement et test

Pour exécuter Elasticsearch, utilisez la commande suivante (l'initialisation peut prendre un peu de temps) :

```
sudo systemctl start elasticsearchCopier
```

Si jamais vous rencontrez des problèmes d'initialisation, veuillez vérifier les logs du service elasticsearch à l'aide de la commande suivante :

```
sudo journalctl -f -u elasticsearchCopier
```

Pour confirmer que tout fonctionne comme prévu, pointez votre commande curl ou votre navigateur sur l'adresse <http://localhost:9200>, et vous devriez voir quelque chose comme la sortie suivante :

```
curl localhost:9200Copier
```

Résultat :

```
{

  "name" : "hatim-linux",

  "cluster_name" : "elasticsearch",
```

```
"cluster_uuid" : "U1zH_yFqTUmdMRckR1gHLQ",

"version" : {

  "number" : "7.8.0",

  "build_flavor" : "default",

  "build_type" : "deb",

  "build_hash" : "757314695644ea9a1dc2fec26d1a43856725e65",

  "build_date" : "2020-06-14T19:35:50.234439Z",

  "build_snapshot" : false,

  "lucene_version" : "8.5.1",

  "minimum_wire_compatibility_version" : "6.8.0",

  "minimum_index_compatibility_version" : "6.0.0-beta1"

},

"tagline" : "You Know, for Search"

}
```

Pour initialiser le service à chaque démarrage de la machine, lancez la commande suivante :

```
sudo systemctl enable elasticsearchCopier
```

kibana

Installation

Puisque nous avons déjà défini le dépôt dans le système, tout ce que nous avons à faire pour installer kibana est d'exécuter la commande suivante:

Sous la famille debian :

```
sudo apt-get install kibanaCopier
```

Sous la famille redhat :

```
sudo yum install kibanaCopier
```

configuration

Le fichier de configuration de kibana se retrouve dans `/etc/kibana/kibana.yml`. Si jamais vous avez modifié avec ce fichier, assurez-vous juste que la configuration kibana possède les bonnes informations pour communiquer avec Elasticsearch :

```
elasticsearch.hosts: ["http://localhost:9200"]
```

Lancement et test

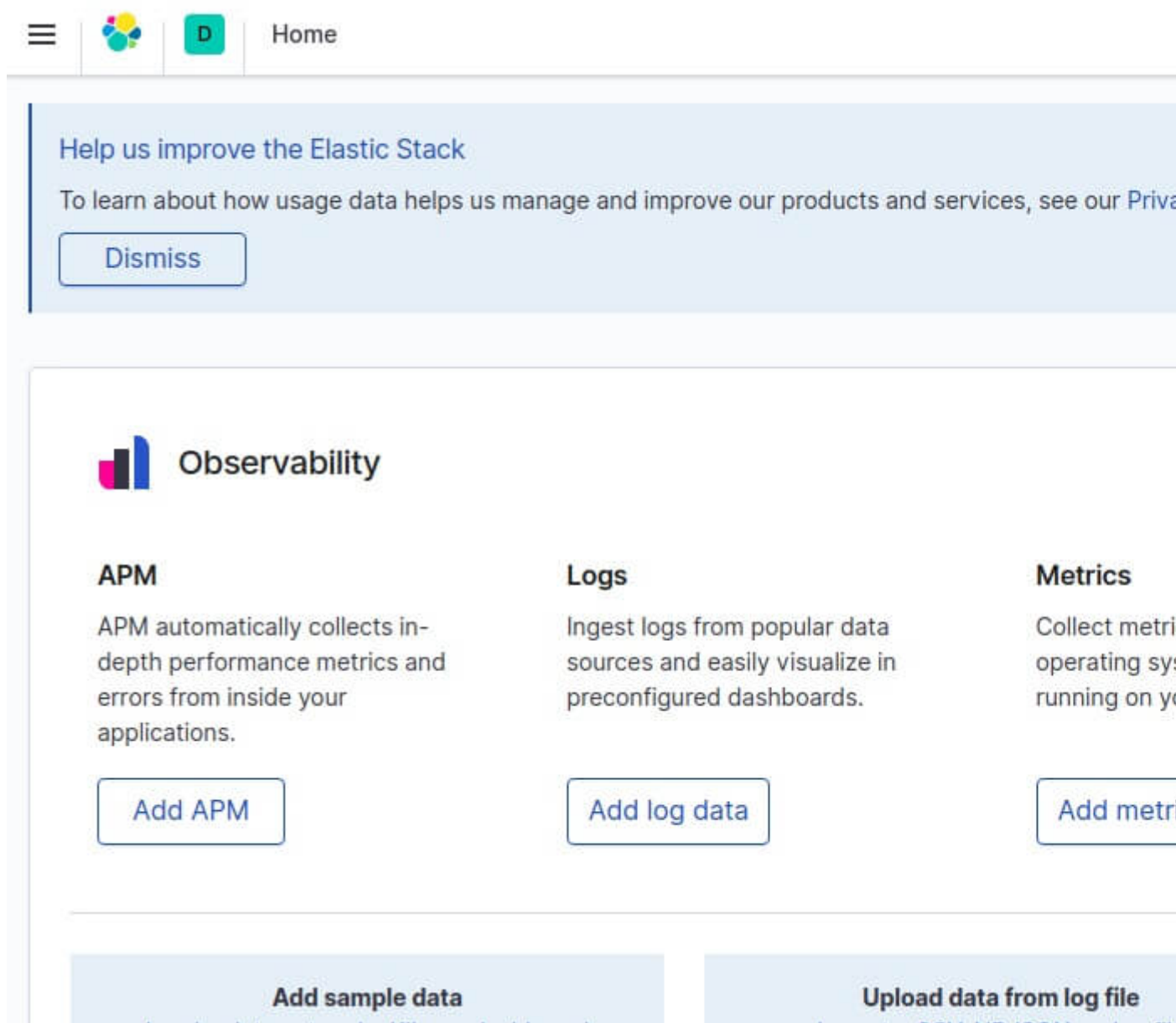
Voici la commande pour démarrer Kibana :

```
sudo systemctl start kibanaCopier
```

Si jamais vous rencontrez des problèmes d'initialisation, veuillez vérifier les logs du service kibana comme suit :

```
sudo journalctl -f -u kibanaCopier
```

Pour tester Kibana, ouvrez dans votre navigateur l'url <http://localhost:5601> afin de voir la page d'accueil Kibana :



Pour initialiser le service Kibana à chaque démarrage de la machine, lancez la commande suivante :

```
sudo systemctl enable kibana
```

Logstash

Installation

Logstash nécessite au minimum la version 8 de java pour fonctionner, nous allons donc commencer le processus de configuration de Logstash avec:

Sous la famille debian :

```
sudo apt-get install default-jreCopier
```

Sous la famille redhat, java 8 :

```
yum install java-11-openjdk.x86_64

# ou java 8

yum install java-1.8.0-openjdk.x86_64Copier
```

Enfin, vérifiez que java est installé:

```
java -versionCopier
```

Résultat :

```
openjdk version "11.0.7" 2020-04-14
...
```

Comme pour kibana, puisque nous avons déjà défini le dépôt dans le système, tout ce que nous avons à faire pour installer Logstash est d'exécuter:

Sous la famille debian :

```
sudo apt-get install logstashCopier
```

Sous la famille redhat :

```
sudo yum install logstashCopier
```

Configuration

Le fichier de configuration de Logstash est le suivant : `/etc/logstash/logstash.yml` et permet de configurer des paramètres généraux comme par exemple le nom du nœud, le port, le niveau des logs etc... Pour ce cours nous laisserons la configuration par défaut.

Lancement et test

Voici la commande pour démarrer logstash :

```
sudo systemctl start logstashCopier
```

Si jamais vous rencontrez des problèmes d'initialisation, vérifiez les logs du service Logstash comme suit :

```
sudo journalctl -f -u logstashCopier
```

Pour initialiser le service à chaque démarrage de la machine, lancez la commande suivante :

```
sudo systemctl enable logstashCopier
```

Pour tester votre installation Logstash, vous devez configurer un pipeline de données. Nous aborderons cette partie dans le prochain chapitre.

Utilisation de la stack ELK sur les logs Apache

Dans ce chapitre, nous allons apprendre à utiliser la stack ELK en analysant en temps réel les logs d'accès Apache.

Introduction

Après avoir installé la suite ELK dans l'[article précédent](#). Nous allons dans cet article pratiquer et s'**initier à l'utilisation des différents composants de la stack ELK en analysant en temps réel les logs d'accès Apache.**

Prérequis

Pour bien suivre cet article, vous devez au préalable installer et démarrer le package d'Apache sur votre machine ELK. Pour cela, lancez les commandes suivantes :

Pour la famille Debian :

```
sudo apt-get install -y apache2Copier
```

Pour la famille Redhat :

```
sudo yum install -y httpdCopier
```

Utilisez l'outil `systemctl` pour démarrer le service Apache :

Pour la famille Debian :

```
sudo systemctl start apache2Copier
```

Pour la famille Redhat :

```
sudo systemctl start httpdCopier
```

Enfin, très important, n'oubliez pas de rajouter l'utilisateur logstash au groupe adm :


```
sudo usermod -aG adm logstashCopier
```

Information

Le groupe d'utilisateurs adm est utilisé pour permettre à ses membres de lire le contenu des fichiers de logs.

Logstash

La première étape à effectuer est le traitement des logs sous forme d'un ou plusieurs pipelines avec l'outil Logstash.

Comment fonctionne de Logstash ? Il est capable d'extraire des données de presque n'importe quelle source de données à l'aide des plugins d'entrée, et d'appliquer une grande variété de transformations et d'améliorations de données à l'aide de plugins de filtre et enfin d'expédier ces données vers un grand nombre de destinations à l'aide de plugins de sortie. Vous comprendrez alors que Logstash joue un rôle très important dans la pile en récupérant, filtrant et expédiant vos données afin de les utiliser plus facilement sur le reste des composants.

Pour utiliser Logstash, il faut créer des fichiers de configuration qui contiendront trois sections principales que nous allons par passer en revue ci-dessous. À savoir que chacune de ces sections est responsable de différentes fonctions et utilisant différents plugins Logstash.

Commencez déjà par créer un fichier `apache.conf` dans le dossier de configuration Logstash situé dans `/etc/logstash/conf.d/` et rajoutez-y les trois sections de configuration principales qui seront pour le moment vides :

```
input {}

filter {}

output {}Copier
```

Entrées Logstash

Les points de départ de toute configuration Logstash sont vos entrées. Les entrées sont des plugins Logstash responsables de la récupération des données de différentes sources de données. Vous trouverez l'intégralité des plugins d'entrée [ici](#). Pour notre exemple, nous avons besoin de récupérer les données depuis un fichier donc nous utiliserons le [plugin d'entrée file](#), comme suit :

Pour la famille debian :

```
input {  
  
    file { path => "/var/log/apache2/access.log" }  
  
}  
  
...Copier
```

Pour la famille redhat :

```
input {  
  
    file { path => "/var/log/httpd-access.log" }  
  
}  
  
...Copier
```

En regardant la documentation du [plugin d'entrée file](#), vous remarquerez que vous pouvez utiliser différents paramètres pour cette entrée. Comme par exemple, le paramètre `start_position` qui permet de choisir où Logstash doit commencer sa lecture initiale des fichiers, soit au début ou à la fin. La valeur par défaut est `end` qui traite les fichiers comme des flux direct en temps réel (comme la commande `tail -f`) et comme son nom l'indique, la lecture débutera à partir de la fin du fichier, mais si vous souhaitez lire un fichier depuis le début, forcez alors la valeur à `beginning` :

```
input {
```

```
file {  
  
    path => "/var/log/apache2/access.log"  
  
    start_position => "beginning"  
  
    sincedb_path => "/dev/null"  
  
}  
  
}  
  
...Copier
```

Par défaut, logstash écrit la dernière position de lecture dans un fichier log qui réside généralement dans `$HOME/.sincedb` ou dans le dossier `/var/lib/logstash/plugins/inputs/file/`. Afin de forcer la relecture complète d'un fichier à chaque exécution de logstash, il faut changer le chemin du fichier de logs de position en valorisant `sincedb_path` à `/dev/null`. Ou sinon vous pouvez supprimer uniquement la ligne correspondante de votre fichier `.sincedb` et redémarrez le service logstash.

Filtres Logstash

Les filtres sont des modules qui vont traiter les données récupérées depuis l'input en s'aidant lui aussi de [différents plugins de filtrage](#). Pour notre exemple, nous utiliserons le plugin `grok` qui vous permet d'analyser facilement des données de logs non structurées en quelque chose de plus exploitable en utilisant des expressions régulières. Cet outil est parfait pour tout format de logs généralement écrit pour les humains comme les logs applicatifs (ex: mysql, apache, nginx, etc ...).

La syntaxe d'un pattern grok est la suivante :

```
%{SYNTAX:SEMANTIC}
```

- **SYNTAX** : nom du pattern qui correspond à un filtre prédéfini, comme par exemple le pattern **NUMBER** qui prend en compte des nombres tel quel 3 ou 3.55 ou le pattern **IP** qui correspond à une IP comme 192.168.1.20.
- **SEMANTIC** est l'identifiant de la valeur récupérée par votre **SYNTAX**.

Par exemple pour traiter la ligne de log suivante :

```
55.3.244.1 GET /index.html 15824 0.043
```

Le pattern grok sera le suivant :

```
filter {  
  
  grok {  
  
    match => { "message" => "%{IP:client} %{WORD:method}  
%{URIPATHPARAM:request} %{NUMBER:bytes} %{NUMBER:duration}" }  
  
  }  
  
}
```

Copier

Après traitement du filtre grok, nous obtiendrons les champs suivants :

```
client: 55.3.244.1  
  
method: GET  
  
request: /index.html  
  
bytes: 15824  
  
duration: 0.043
```

Pour notre besoin, nous allons plutôt découvrir et utiliser des patterns APACHE grok pré-configurés. Vous pouvez utiliser soit le pattern **COMMONAPACHELOG** qui récupère des informations de base (IP source, code HTTP, etc ...) ou le pattern **COMBINEDAPACHELOG** qui récupère des informations supplémentaires comme le user-agent.

Information

Pour les plus curieux d'entre vous qui souhaitent connaître le pattern complet de **COMMONAPACHELOG** et **COMBINEDAPACHELOG**, regardez la ligne sous le commentaire "# Log formats" sur les [sources de l'APACHE pattern](#).

Pour tester vos patterns, vous pouvez soit utiliser un plugin de sortie que nous verrons dans la section suivante, ou bien utilisez un débogueur grok en ligne, comme [grokdebug](#). Pour utiliser

l'exactitude d'exécution de votre pattern sur votre débogueur, vous n'avez qu'à copier quelques lignes de vos logs apaches :

```
# famille Debian

tail /var/log/apache2/access.log

# famille Redhat

tail /var/log/httpd-access.logCopier
```

Résultat :

```
:::1 - - [30/Jun/2020:16:00:47 +0200] "GET / HTTP/1.1" 200 422 "-"
"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/83.0.4103.116 Safari/537.36"

:::1 - - [30/Jun/2020:16:04:30 +0200] "GET / HTTP/1.1" 200 423 "-"
"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/83.0.4103.116 Safari/537.36"
```

Une fois vos lignes de logs copiées, collez les dans votre débogueur avec votre pattern du plugin grok (sans les doubles guillemets), exemple :

```
:::1 - - [30/Jun/2020:16:00:47 +0200] "GET / HTTP/1.1" 200 422 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36"
:::1 - - [30/Jun/2020:16:04:30 +0200] "GET / HTTP/1.1" 200 423 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36"
```

```
%{COMBINEDAPACHELOG}
```

☐ Add custom patterns ☐ Keep Empty Captures ☒ Named Captures Only ☐ Singles

Vous obtiendrez ainsi le résultat suivant :

```
{
  "clientip": [
    [
      ":::1"
    ]
  ],
  "ident": [
    [
      "_"
    ]
  ],
  "auth": [
    [
      "_"
    ]
  ],
  "timestamp": [
    [
      "30/Jun/2020:16:00:47 +0200"
    ]
  ],
  "verb": [
    [
      "GET"
    ]
  ],
  "request": [
```

Cool notre pattern grok fonctionne! Revenons alors maintenant sur notre fichier `apache.conf` et complétons la section filtre, comme suit :

```
input {

  file { path => "/var/log/apache2/access.log" }

}

filter {

  grok {

    match => { "message" => "%{COMBINEDAPACHELOG}" }

  }

}
```

```

    }

    date {
        match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]
    }

    mutate {
        convert => {
            "response" => "integer"
            "bytes" => "integer"
        }
    }
}

```

Copier

J'ai fait exprès de rajouter deux nouveaux filtre [date](#) et [mutate](#), car ils nous serviront plus tard pour nos visualisations sur Kibana :

Par défaut, logstash basera son horodatage sur l'heure à laquelle l'entrée du fichier de log a été lue et non sur l'horodatage fourni par le fichier de logs Apache. D'où l'utilisation du filtre de [date](#) qui basera son horodatage sur les dates des champs filtrés par grok, soit l'horodatage réel fourni par le fichier de log Apache.

J'utilise également le filtre [mutate](#) avec l'action [convert](#) afin de convertir la réponse HTTP et la taille de la requête qui sont par défaut en type string vers le type entier, car en effet, Kibana gère ses visualisations différemment selon le type de données que vous lui envoyez.

Sorties Logstash

Les points d'arrivée de toute configuration Logstash sont vos sorties, lui-même utilise [différents plugins de sortie](#) qui envoient les données traitées par la phase de filtrage à une destination particulière (ex: elasticsearch). Par ailleurs, les sorties sont la dernière étape du pipeline Logstash.

Pour cet article, je vais vous dévoiler **comment récupérer les événements sur la sortie standard** avec le [plugin de sortie stdout](#) avant de les envoyer plus tard à elasticsearch. Le plugin stdout propose différents formats de sortie, dans notre exemple nous utiliserons le format **rubydebug**, comme ceci :

```
input {  
  
  file { path => "/var/log/apache2/access.log" }  
  
}  
  
filter {  
  
  grok {  
  
    match => { "message" => "%{COMBINEDAPACHELOG}" }  
  
  }  
  
  date {  
  
    match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]  
  
  }  
  
  mutate {  
  
    convert => {  
  
      "response" => "integer"  
  
    }  
  
  }  
  
}
```



```

        "bytes" => "integer"

    }

}

output {

    stdout { codec => rubydebug }

}

```

Debug

Dans cette partie je souhaite vous montrer comment vous pouvez debug votre pipeline Logstash afin d'utiliser du mieux ce plugin, pour cela il faut commencer par stopper le service logstash :

```
sudo systemctl stop logstashCopier
```

Ensuite, nous allons **vérifier la syntaxe de notre code Logstash** avec le binaire logstash situé dans `/usr/share/logstash/bin/logstash` en utilisant l'option `-t` ou l'option `--config.test_and_exit` :

Attention

Selon le type d'installation que vous avez choisi, le binaire logstash peut se trouver dans un autre dossier.

```
sudo /usr/share/logstash/bin/logstash --path.settings /etc/logstash
-f /etc/logstash/conf.d/apache.conf -tCopier
```

Résultat :

```
...

Configuration OK
```

```
[2020-06-30T16:58:12,834][INFO ][logstash.runner] ... Config
Validation Result: OK. Exiting Logstash
```

Information

Les différents options du binaire logstash sont disponibles [ici](#).

Une fois la syntaxe validée, relancez la commande sans l'option `-t` mais avec l'option `--debug`:

```
sudo /usr/share/logstash/bin/logstash --debug --path.settings
/etc/logstash -f /etc/logstash/conf.d/apache.confCopier
```

Visitez ensuite depuis votre navigateur la page d'accueil <http://localhost/> et revenez sur la sortie standard de votre logstash pour observer le résultat suivant :

```
{
  "host" => "hatim-linux",
  "ident" => "-",
  "message" => "::-1 - - [30/Jun/2020:17:07:07 +0200] \"GET /
HTTP/1.1\" 200 423 \"-\" \"Mozilla/5.0 (X11; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116
Safari/537.36\"",
  "bytes" => "423",
  "agent" => "\"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36\"",
  "response" => "200",
  "timestamp" => "30/Jun/2020:17:07:07 +0200",
  "referrer" => "\"-\"",
  "request" => "/",
  "verb" => "GET",
  "path" => "/var/log/apache2/access.log",
```

```
"@timestamp" => 2020-06-30T15:07:07.000Z,  
  
"clientip" => "::1",  
  
"auth" => "-",  
  
"httpversion" => "1.1",  
  
"@version" => "1"  
  
}
```

Passons maintenant à la communication avec elasticsearch.

ElasticSearch

Modifions ensuite notre fichier `apache.conf` pour communiquer avec elasticsearch :

```
input {  
  
    file { path => "/var/log/apache2/access.log" }  
  
}  
  
filter {  
  
    grok {  
  
        match => { "message" => "%{COMBINEDAPACHELOG}" }  
  
    }  
  
    date {  
  
        match => [ "timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ]  
  
    }  
  
}
```

```

mutate {

  convert => {

    "response" => "integer"

    "bytes" => "integer"

  }

}

output {

  elasticsearch {

    hosts => "localhost:9200"

    index => "apache-%{+YYYY.MM.dd}"

  }

}
}Copier

```

Vous remarquerez que j'utilise un index dans le bloc de code de sortie **elasticsearch** car c'est le moyen pour elasticsearch de structurer ses données et de les gérer plus rapidement. Le nommage utilisé dans notre index actuel est dynamique grâce au pattern **%{+YYYY.MM.dd}**, ce qui aura pour effet de créer un index par jour, permettant à l'administrateur de facilement s'y retrouver en affichant par exemple que les logs d'une date précise dans Kibana, supprimer que les indexs d'une plage de dates , etc ...

N'oubliez pas de démarrer votre service logstash avec la commande suivante :

```
sudo systemctl start logstashCopier
```

Après démarrage de votre service, logstash va lancer votre pipeline. Ensuite vous pouvez vérifier si l'index s'est correctement créé, pour cela appelez l'API REST fournie par elasticsearch, comme suit :

```
curl "localhost:9200/_cat/indices?v" Copier
```

Résultat :

```
yellow open apache-2020.06.30
```

Kibana

Index Patterns

Rendez-vous ensuite sur Kibana depuis l'url <http://localhost:5601/>. Pour visualiser notre index, allez sur le menu à gauche et cliquez sur "Stack Management" :



Home



Home

Recently viewed



No recently viewed items



Kibana



Discover

Dashboard

Canvas

Maps

Machine Learning

Visualize



Observability



Logs

Metrics

APM

Uptime



Security



SIEM

Management



Dev Tools

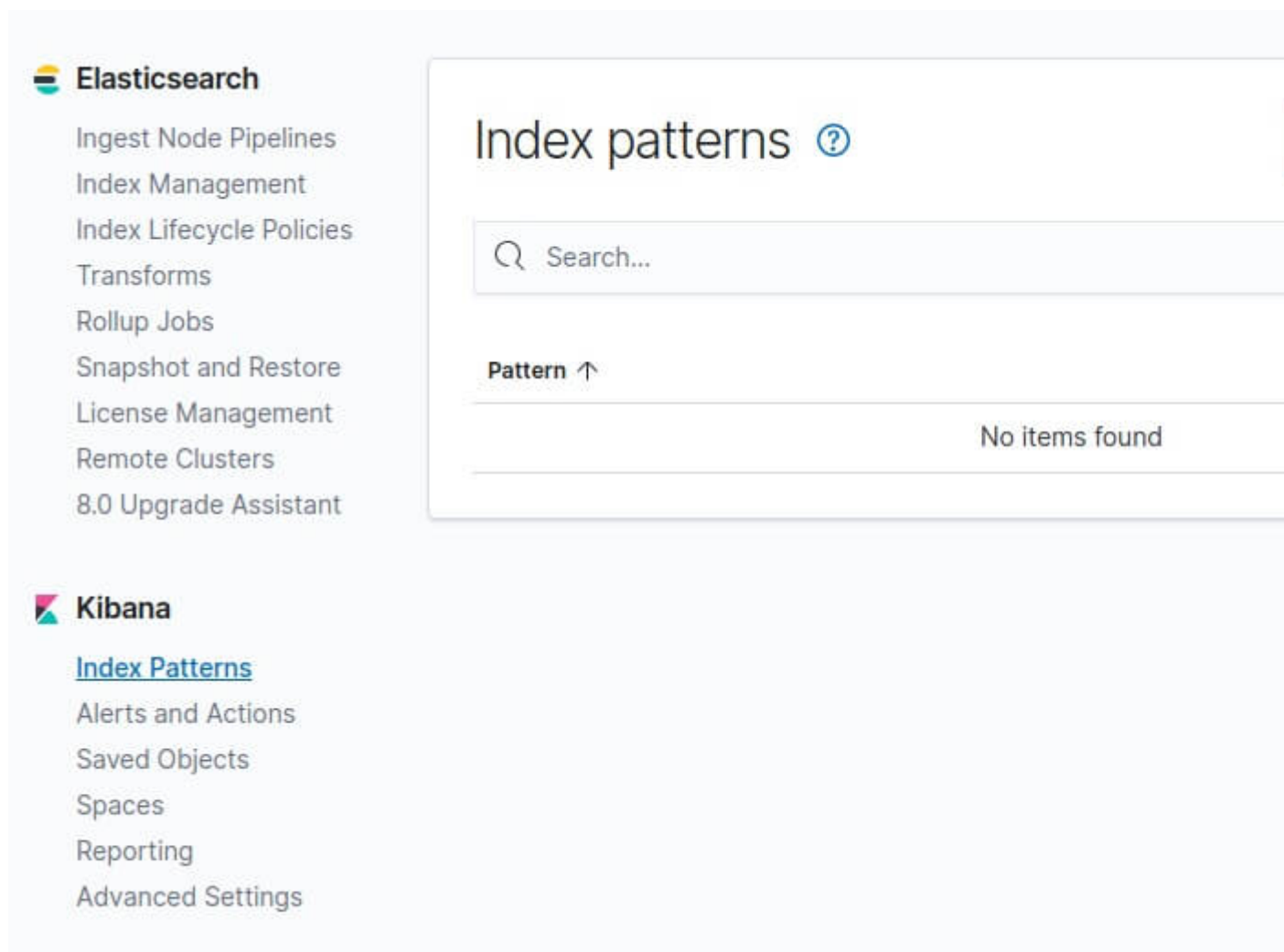
Stack Monitoring

Stack Management



Dock navigation

Par la suite, vous devez **ajouter un pattern index dans kibana** afin de prendre en considération vos index quotidiens Apache récupérés par elasticsearch. Cliquez sur "Index Patterns" sur le volet à gauche et cliquez sur le bouton "Create index pattern" :



Dans notre cas le préfix de nos indexs apache est "apache-", donc le pattern index à créer dans kibana sera **apache-*** :

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

☐ Include system indices

Step 1 of 2: Define index pattern

Index pattern

apache-*

You can use a * as a wildcard in your index pattern.
You can't use spaces or the characters \, /, ?, ", <, >, |.

> Next step

✓ **Success!** Your index pattern matches **1 index**.

apache-2020.07.01

Rows per page: 10 ▾

Cliquez ensuite sur "Next Step". Ensuite il nous demande par quel moyen il doit gérer le filtre temporel (timestamp). Ce filtre est utile pour filtrer et affiner nos données par plage horaire. Nous avons prévu le coup sur notre configuration logstash en créant un champ `@timestamp`, et c'est celui là qu'on sélectionnera :

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

☐ Include system indices

Step 2 of 2: Configure settings

You've defined **apache-*** as your index pattern. Now you can specify some settings before we create it.

Time Filter field name [Refresh](#)

@timestamp

The Time Filter will use this field to filter your data by time. You can choose not to have a time field, but you will not be able to narrow down your data by a time range.

[Show advanced options](#)

[Back](#)

Create index pattern

Enfin, cliquez sur le bouton "Create index pattern" et vous verrez apparaître tous vos champs :

Time Filter field name: '@timestamp'

Default

This page lists every field in the **apache-*** index and the field's associated core type as recorded by Elasticsearch. To change a field type, use the Elasticsearch [Mapping API](#)

Fields (36)

Scripted fields (0)

Source filters (0)

All field types ▾

Name	Type	Format	Search...	Aggreg...	Excluded
@timestamp 🕒	date		●	●	
@version	string		●		
@version.keyword	string		●	●	
_id	string		●	●	
_index	string		●	●	
_score	number				
_source	_source				
_type	string		●	●	
agent	string		●		
agent.keyword	string		●	●	

Rows per page: 10 ▾

< **1** 2 3 4 >

>

Découvrir vos logs

Pour découvrir vos logs, sur le menu à gauche cliquez sur Discover :



Home



Home

Recently viewed



No recently viewed items



Kibana



Discover

Dashboard

Canvas

Maps

Machine Learning

Visualize



Observability



Logs

Metrics

APM

Uptime



Security



SIEM

Management



Dev Tools

Stack Monitoring

Stack Management



Dock navigation

Ensuite, Faites quelques visites depuis votre navigateur sur la page d'accueil d'apache <http://localhost/> et revenez sur la page de discover :



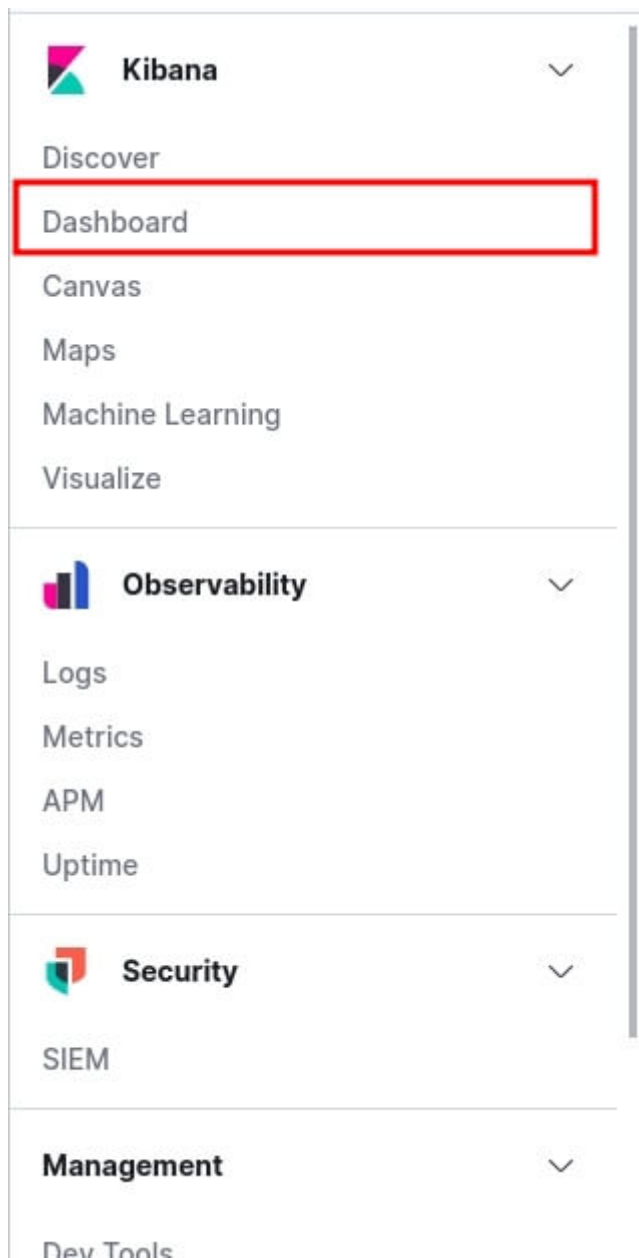
Vous avez également la possibilité de filtrer vos logs par champ depuis la barre de recherche du Discover :

<div> <div> <div></div> <div></div> </div> <div>Search</div> </div>	
<input type="checkbox"/> _id	Filter results that contain <code>_id</code>
<input type="checkbox"/> _index	Filter results that contain <code>_index</code>
<input type="checkbox"/> _type	Filter results that contain <code>_type</code>
<input type="checkbox"/> @timestamp	Filter results that contain <code>@timestamp</code>
<input type="checkbox"/> @version.keyword	Filter results that contain <code>@version.keyword</code>
<input type="checkbox"/> @version	Filter results that contain <code>@version</code>
<input type="checkbox"/> agent.keyword	Filter results that contain <code>agent.keyword</code>
<input checked="" type="checkbox"/> agent	Filter results that contain <code>agent</code>
<input type="checkbox"/> auth.keyword	Filter results that contain <code>auth.keyword</code>
<input type="checkbox"/> auth	Filter results that contain <code>auth</code>
<input type="checkbox"/> bytes.keyword	Filter results that contain <code>bytes.keyword</code>
<input type="checkbox"/> bytes	Filter results that contain <code>bytes</code>
<input type="checkbox"/> clientip.keyword	Filter results that contain <code>clientip.keyword</code>
<input type="checkbox"/> clientip	Filter results that contain <code>clientip</code>
<input type="checkbox"/> host.keyword	Filter results that contain <code>host.keyword</code>
<input type="checkbox"/> host	Filter results that contain <code>host</code>
<input type="checkbox"/> httpversion.keyword	Filter results that contain <code>httpversion.keyword</code>
<input type="checkbox"/> httpversion	Filter results that contain <code>httpversion</code>

Exemple : "response : 404" pour n'afficher que les requêtes en erreur 404.

Dashboard

L'étape suivante est de **créer un tableau de bord afin de visualiser une collection de visualisations en temps réel**. Pour commencer, ouvrez le menu, accédez à Dashboard , puis cliquez sur "Create dashboard" :



Pour ajouter des éléments à votre dashboard vous devez **créer des visualisations Kibana** que vous pouvez déplacer et redimensionner dans votre dashboard. Vous pouvez ajouter des visualisations à partir de plusieurs index patterns et la même visualisation peut apparaître dans plusieurs tableaux de bord.

Dans notre exemple nous allons commencer par créer une visualisation qui permet d'afficher le nombre total d'utilisateurs unique. Pour ce faire, créez une visualisation en cliquant sur "Create new" et dans la fenêtre de "New visualisation" nous allons choisir le type de visualisation "Metric" :

New Visualization

Filter



Lens



Area



Controls



Data Table



Gauge



Goal



Heat Map



Horizontal Bar



Line



Maps



Markdown



Metric



Pie



TSVB



Tag Cloud



Timelion



Vega



Vertical Bar

Select a visualization type

Start creating your visualization by selecting a type for that visualization.

Try Lens, our new, intuitive way to create visualizations.

[Go to Lens](#)

Pour information, voici la **liste des types de visualisations Kibana les plus fréquemment utilisées** :

- **Line, area, et bar charts** : compare différentes métriques sur l'axe X et Y.
- **Pie chart** : graphique circulaire.
- **Data table** : données en format de tableau.
- **Metric** : affiche une seule métrique.
- **Goal and gauge** : affiche un nombre avec des indicateurs de progression.
- **Tag cloud** : affiche les mots dans un nuage, où la taille du mot correspond à son importance.

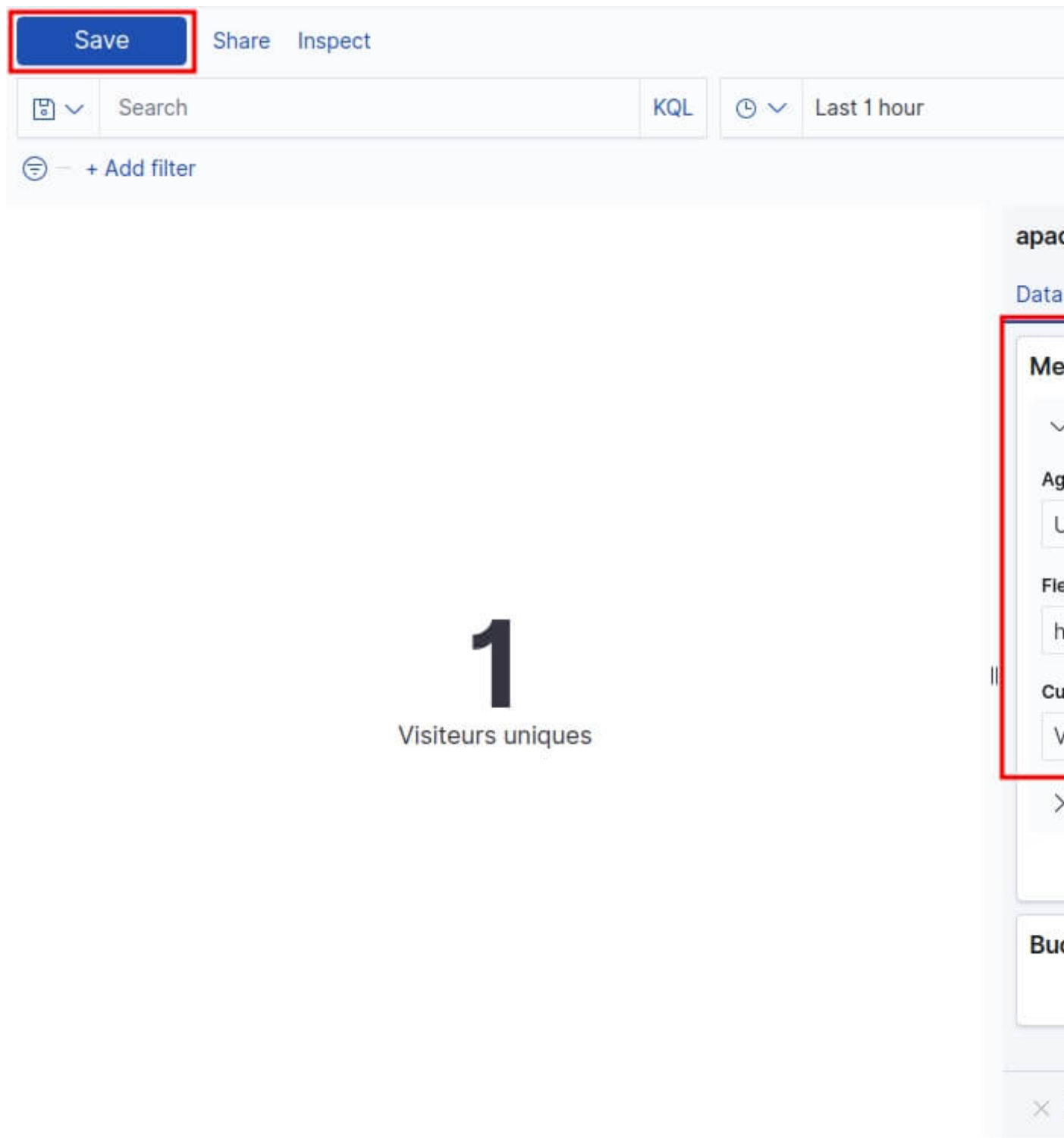
On vous demande ensuite de **paramétrer votre visualisation**, vous devez choisir d'abord votre agrégation qui correspond aux métriques extraites pour générer des valeurs de données. Voici les valeurs les plus communes :

- **Average** : valeur moyenne.
- **Count** : nombre total de documents correspondant à une requête.
- **Max** : la valeur la plus élevée.
- **Median** : médiane.
- **Min** : la valeur la plus basse.
- **Sum** : La valeur totale.
- **Unique Count** : nombre unique d'une métrique.

Information

Dans le langage Elasticsearch, un document correspond aux données JSON sérialisées.

Dans notre cas, nous utiliserons l'agrégation **Unique Count** en utilisant le champ **host** :



Cliquez ensuite sur "Save" et choisissez le nom de votre visualisation.

Rajoutons un autre graphique qui permet d'afficher la taille moyenne des requêtes temporellement. Pour ce type de besoin nous aurons besoin d'une visualisation de type "area". Pour l'axe Y nous allons utiliser une agrégation de type "Average" sur le champ **byte** :

Metrics

Y-axis

Aggregation

Average

Average help

Field

bytes

Custom label

bytes

> Advanced

+ Add

Pour l'axe X ça sera un peu différent car nous utiliserons les Bucket aggregations qui trient les documents en compartiments selon le contenu du document. Voici les valeurs les plus communes :

- **Date histogram** : fractionne un champ de date en compartiments par intervalle.
- **Date range** : valeurs comprises dans une plage de dates que vous spécifiez.
- **Filter** : filtre les données récupérées (ex : traiter que les erreurs 404).
- **IPv4 range** : plages d'adresses IPv4.
- **Range** : plages de valeurs pour un champ numérique.
- **Terms** : Spécifiez le nombre d'éléments supérieurs ou inférieurs d'un champ donné à afficher, classés par nombre ou par une métrique personnalisée.

Pour notre cas nous utiliserons le type "Date histogram" :

Buckets

☒ X-axis

Aggregation

Date Histogram

▼

Field

@timestamp

▼

Minimum Interval

Auto

✕

▼

Select an option or create a custom value. Examples: 30s, 20m, 24h, 2d, 1w, 1M

☐ Drop partial buckets

Custom label

> Advanced

+

 Add

Un dernier graphique avant de clôturer cet article. Nous allons cette fois-ci afficher le top des requêtes en erreur sous forme d'un tableau. Créez une nouvelle visualisation de type "Data table" avec comme configuration une agrégation de type "Count" (par défaut) et une Bucket aggregation de type "Terms" sur le champ "request" et trier par ordre décroissant par l'agrégation "Count", ce qui nous affichera pour le moment que les pages web les plus visitées. Cette partie de la configuration ressemblera à ceci :

apache-*

Data

Options

▼ Metric

Aggregation

Count

Count help

Custom label

> Advanced

+ Add

Buckets

▼ Split rows

⊞ ✕

Aggregation

Terms

Terms help

Field

request.keyword

Order by

Metric: Count

Order

Descending

Size

5

☐ Group other values in separate bucket

☐ Show missing values

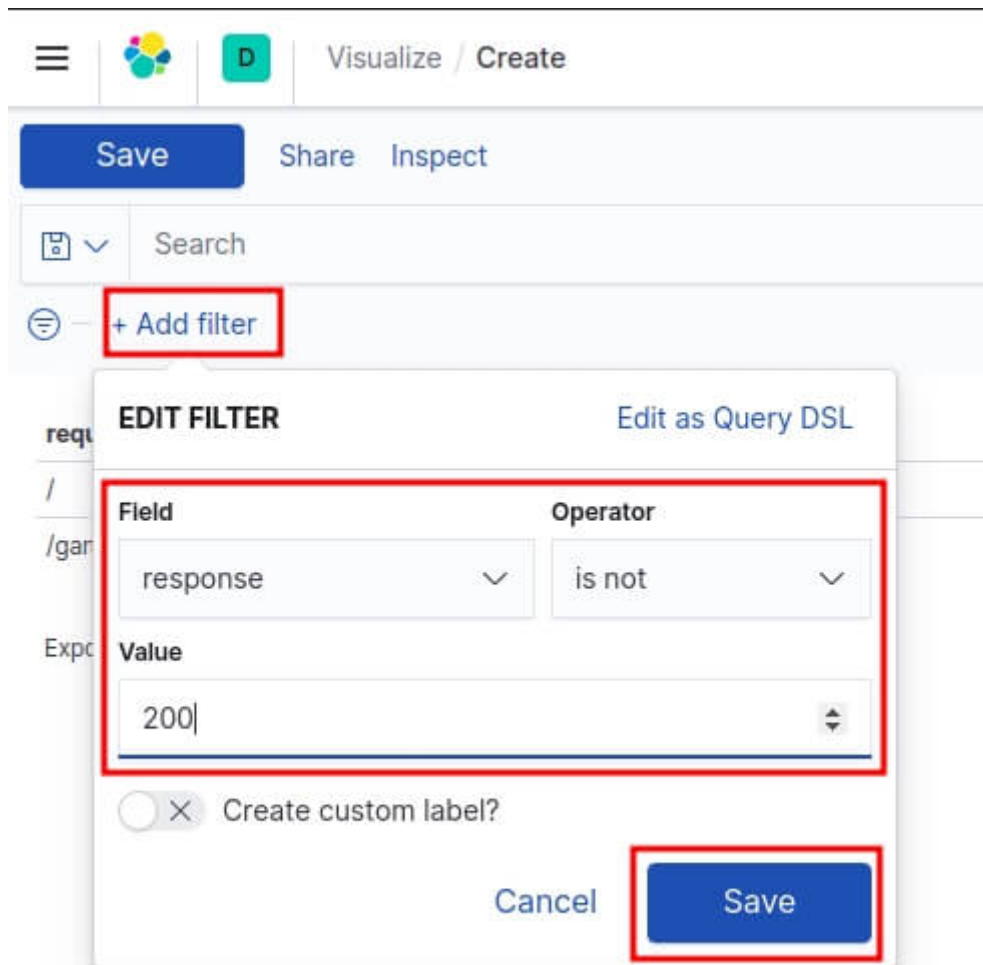
Custom label

> Advanced

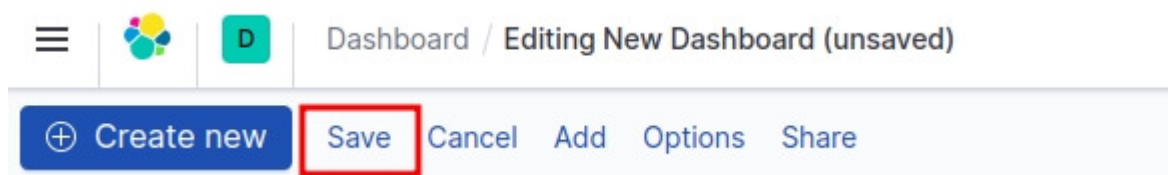
✕ Discard

▶ Update

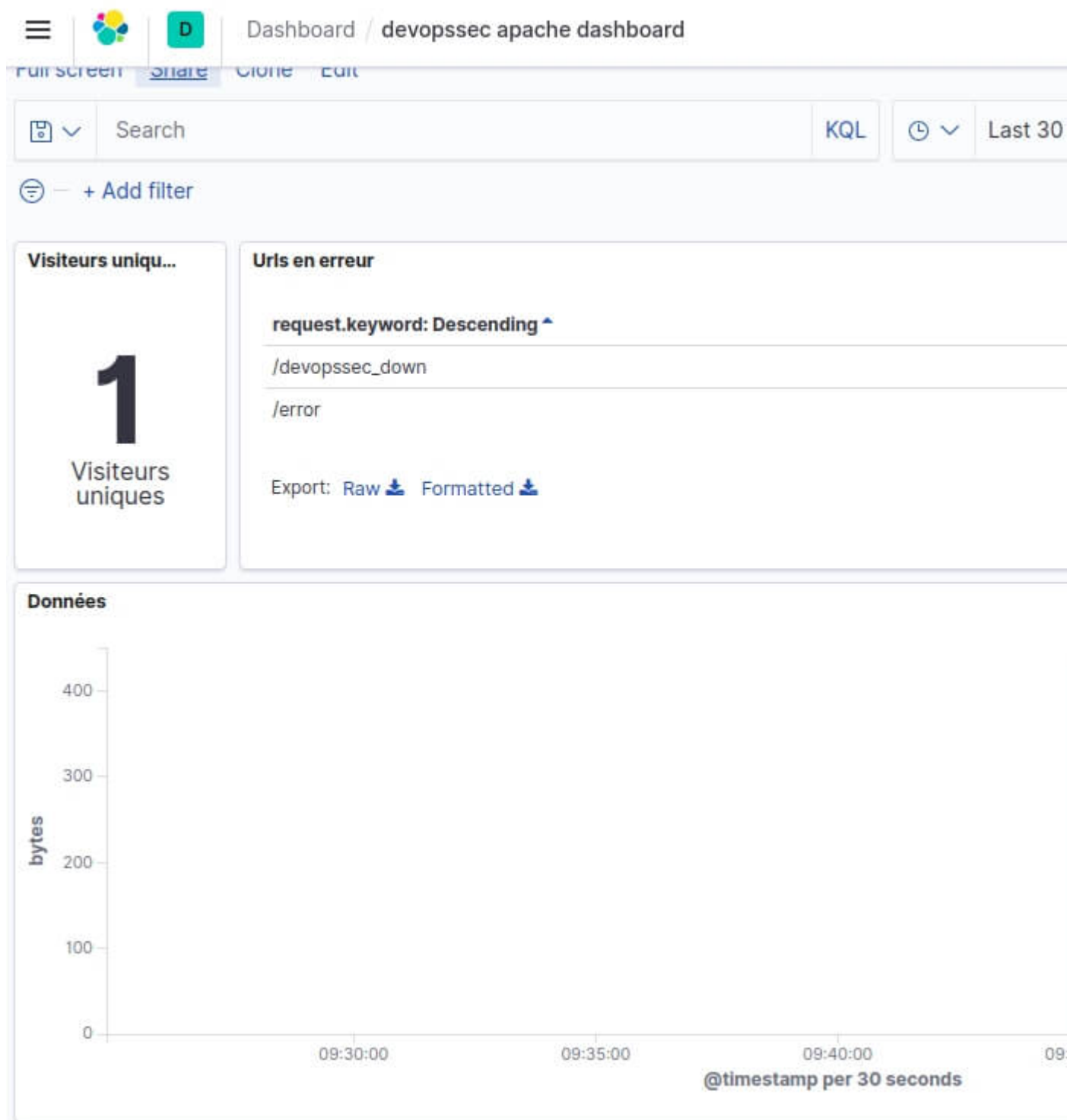
Ensuite pour récupérer que les requêtes en erreur, nous filtrerons ces requêtes si elles ont une réponse différente au code HTTP 200. Pour cela, vous devez cliquer sur le bouton situé en haut à gauche nommée "+ Add filter" et ajouter le filtre suivant :



Lorsque vous avez terminé d'ajouter et d'organiser les panneaux, enregistrez le tableau de bord. Dans la barre d'outils Kibana, cliquez sur "Save" et saisissez ensuite le titre du tableau de bord et la description facultative , puis enregistrez votre tableau de bord.



Le dashboard final ressemble à ceci :



Conclusion

Dans ce chapitre nous avons appris à utiliser plus en profondeur la suite ELK en partant d'un simple fichier de logs Apache à des données visuelles graphiques en temps réels que vous

pouvez manipuler comme vous le souhaitez démontrant la puissance d'ELK et l'utilité d'une solution de centralisation de journalisation.

Comprendre et utiliser Filebeat dans la stack ELK

Dans ce chapitre, nous allons apprendre à utiliser Filebeat dans la suite ELK en analysant les logs Apache.

Introduction

Tout comme Logstash, Filebeat peut être utilisé pour envoyer des logs à partir d'une source de données basée sur des fichiers vers une destination de sortie prise en charge. Mais la comparaison s'arrête là. Car dans la plupart des cas, nous utiliserons les deux en tandem lors de la création de notre pipeline de journalisation avec la pile ELK, puisque les deux ont une fonction différente.

La limite de Logstash

Logstash a été développé à l'origine par [Jordan Sissel](#) pour gérer le streaming d'une grande quantité de données de logs provenant de plusieurs sources, et après que Sissel a rejoint l'équipe Elastic, Logstash est passé d'un outil autonome à une partie intégrante de la pile ELK (Elasticsearch, Logstash, Kibana).

Pour pouvoir déployer un système de journalisation centralisé efficace, un outil capable à la fois d'extraire des données de plusieurs sources de données et de leur donner un sens est nécessaire. Tel est le rôle joué par Logstash, il gère les tâches de réception des données provenant de plusieurs systèmes, les transformants en un ensemble significatif de champs et éventuellement en continu la sortie vers une destination définie pour le stockage.

Eh bien, il y avait, et il y a toujours, un problème en suspens avec Logstash, et c'est **la performance**. Logstash nécessite la JVM (Java Virtual Machine) pour s'exécuter, et cette dépendance couplée à l'implémentation dans Ruby est devenue la cause première d'une consommation de mémoire importante, en particulier lorsque plusieurs pipelines et qu'un filtrage avancé sont impliqués.

La naissance de Beats

Lumberjack a été initialement développé comme une expérience d'externalisation des tâches d'extraction de données et était destiné à être utilisé comme un **expéditeur léger pour collecter les logs** avant de les envoyer pour traitement sur une autre plate-forme (telle que Logstash ou directement sur Elasticsearch). Écrit en Go, le concept derrière Lumberjack était de développer un protocole réseau qui serait plus efficace pour gérer de gros volumes de données, aurait une **faible empreinte mémoire** et prendrait en charge le chiffrement.

Le projet a été renommé ensuite en "**Logstash-Forwarder**", constituant et incluant désormais le protocole réseau et le programme de journalisation. Ensuite, une deuxième version du protocole Lumberjack a été développée, dépréciant ainsi Logstash-Forwarder. Ce nouveau protocole connu sous le nom de "Beats" est utilisé par une nouvelle famille d'expéditeurs (ex d'agent Beats : **Filebeat** pour les fichiers logs, **Packetbeat** pour les métriques réseaux **Metricbeat**, etc ...).

Dans ce chapitre nous allons **apprendre à utiliser Filebeat**.

Dois-je utiliser Filebeat ou Logstash ?

Pour collecter des logs sur des machines distantes, Filebeat est recommandé car il nécessite moins de ressources qu'une instance de Logstash. Cependant, vous utiliseriez Logstash si vous souhaitez manipuler vos logs pour ajouter ou supprimer des champs ou enrichir vos données à l'aide de filtres d'entrée/sortie pour les envoyer ensuite à Elasticsearch. Quant à Filebeat, c'est un choix parfait pour récupérer une donnée déjà traitée et la transmettre directement à Elasticsearch.

Pour information, **Filebeat et Logstash peuvent être utilisés conjointement**. Par exemple si vous devez collecter des logs à partir de machines distantes, vous pouvez utiliser Filebeat pour les récupérer et ensuite les envoyer à Logstash si vous voulez faire des transformations sur vos données avant de les envoyer à Elasticsearch.

Utilisation de Filebeat

Installation de Filebeat

Avant de commencer l'installation, assurez-vous d'avoir installé Elasticsearch pour stocker et rechercher nos données, et d'avoir installé Kibana pour les visualiser et les gérer (mon tuto d'installation est disponible [ici](#)).

Il existe plusieurs façons d'installer Filebeat. Dans notre cas nous allons **installer Filebeat depuis le gestionnaire de paquets par défaut** de notre distribution. Pour ce faire, vous devrez d'abord mettre à niveau votre système et vos paquets:

```
sudo apt update -y && sudo apt upgrade -yCopier
```

Pour les machines appartenant à la famille debian, vous devrez peut-être installer le paquet apt-transport-https avant de continuer:

```
sudo apt-get install apt-transport-httpsCopier
```

Téléchargez et installez ensuite la clé de signature publique:

Sous la famille debian:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -Copier
```

Sous la famille redhat:

```
sudo rpm --import https://packages.elastic.co/GPG-KEY-elasticsearchCopier
```

L'étape suivante consiste à ajouter le dépôt Elastic sur votre système :

Sous la famille debian:

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.listCopier
```

Sous la famille redhat, créez un fichier et nommez le par exemple *elastic.repo* dans le répertoire */etc/yum.repos.d/*, contenant:

```
[elastic-7.x]
```

```
name=Elastic repository for 7.x packages
```

```
baseurl=https://artifacts.elastic.co/packages/7.x/yum
```

```
gpgcheck=1
```

```
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

```
enabled=1
```

```
autorefresh=1
```

```
type=rpm-mdCopier
```

Il ne vous reste plus qu'à mettre à jour vos référentiels et installer Filebeat:

Sous la famille debian:

```
sudo apt-get update && sudo apt-get install filebeatCopier
```

Sous la famille redhat:

```
sudo yum install filebeatCopier
```

Pour exécuter Filebeat, utilisez la commande suivante:

```
sudo systemctl start filebeatCopier
```

Si jamais vous rencontrez des problèmes d'initialisation, veuillez vérifier les logs du service Filebeat à l'aide de la commande suivante :

```
sudo journalctl -f -u filebeatCopier
```

Configuration de Filebeat

Le fichier de configuration de Filebeat se retrouve dans `/etc/filebeat/filebeat.yml` . Dans ce fichier, assurez-vous bien que la configuration Filebeat possède les bonnes informations pour communiquer avec Kibana et Elasticsearch, si besoin décommentez les lignes suivantes :

```
output.elasticsearch:
```

```
  hosts: ["localhost:9200"]
```

```
  username: "" #(si pas de login/mot de passe ne rien mettre)
```

```
  password: "" #(si pas de login/mot de passe ne rien mettre)
```

```
...
```

```

setup.kibana:

  host: "localhost:5601"

...

# Optionnelle

filebeat.config.modules:

  path: ${path.config}/modules.d/*.yaml

  reload.enabled: true

  reload.period: 15s

```

Ci-dessous une explication détaillée sur **les options de configuration Filebeat** utilisées:

- **filebeat.config.modules** : où nous retrouverons l'option **path** qui est l'emplacement des fichiers de configuration des modules Filebeat situés par défaut dans le dossier `/etc/filebeat/modules.d/`, et pour les cibler par défaut filebeat utilise la regex `*.yaml`. Par ailleurs, nous avons également la possibilité d'activer ou non le rechargement automatique de la configuration qui est par défaut à `false` dans l'option **reload.enabled**. Si vous passez la valeur `true` comme moi, alors Filebeat surveillera périodiquement (ici toutes les 15 secondes) vos fichiers de configuration et si des changements sont détectés, il rechargera l'ensemble de la configuration.
- **setup.kibana** : pour que les tableaux de bord fonctionnent, nous devons spécifier le point de terminaison Kibana. Vous devrez entrer l'URL de votre hôte Kibana et vos informations d'identification (nom d'utilisateur/mot de passe) si nécessaire.
- **output.elasticsearch** : spécifie la sortie à laquelle nous envoyons les métriques Filebeat. Nous utilisons Elasticsearch, vous devrez donc fournir l'hôte, le protocole et les informations d'identification Elasticsearch si nécessaire.

Pour initialiser le service à chaque démarrage de la machine, lancez la commande suivante:

```
sudo systemctl enable filebeat
```

Les modules Filebeat (Apache2)

Les modules Filebeat vous permettent de commencer rapidement à traiter les formats de logs courants. Ils contiennent des configurations par défaut, des définitions de pipeline pour Elasticsearch et des

tableaux de bord préfabriqués Kibana pour vous aider à mettre en œuvre et à déployer une solution de surveillance des logs rapidement. La liste des modules est disponible [ici](#).

Pour cet article, nous allons utiliser le **module apache2** afin d'**analyser les logs d'accès et d'erreurs créés par un serveur Apache**. Lorsque vous exécutez le module, il effectue quelques tâches automatiquement pour vous :

- Définit les chemins des fichiers logs par défaut (vous pouvez remplacer les valeurs par défaut)
- S'assure que chaque événement de logs multiligne est envoyé en tant qu'événement unique
- Analyse et traite les lignes de log Apache automatiquement, et façonne les données dans une structure adaptée à la visualisation dans Kibana
- Déploie des tableaux de bord automatique dans Kibana pour visualiser les données de vos logs

Avant de suivre ces étapes, vérifiez qu'Elasticsearch et Kibana sont en cours d'exécution et qu'Elasticsearch est prêt à recevoir des données de Filebeat:

```
sudo systemctl status elasticsearch kibana
```

Résultat :

• elasticsearch.service - Elasticsearch

```
Loaded: loaded (/lib/systemd/system/elasticsearch.service; disabled; vendor preset: enabled)
```

```
Active: active (running) since Mon 2021-07-12 14:47:17 CEST; 3h 21min ago
```

• kibana.service - Kibana

```
Loaded: loaded (/etc/systemd/system/kibana.service; disabled; vendor preset: enabled)
```

```
Active: active (running) since Mon 2021-07-12 14:46:47 CEST; 3h 21min ago
```

Pour **remplacer les chemins par défaut pour l'accès au serveur HTTP Apache et les logs d'erreurs dans Logstash**. Rendez-vous dans le fichier de configuration du module apache `/etc/filebeat/modules.d/apache.yml`:

```
- module: apache
```

```
access:
```

```
enabled: true
```

```
var.paths: ["/nouveau-path/log/apache/access.log*"]

error:

enabled: true

var.paths: ["/nouveau-path/log/apache/error.log*"]Copier
```

Pour configurer et exécuter le module Apache, lancez la commande suivante:

```
sudo filebeat modules enable apacheCopier
```

Pour voir la liste des modules activés et désactivés, exécutez :

```
sudo filebeat modules listCopier
```

Résultat :

Enabled:

apache

Disabled:

activemq

...

Ensuite lancez la sous-commande `setup` pour charger les tableaux de bord dans Kibana (la commande peut prendre un peu de temps pour se terminer):

```
sudo filebeat setupCopier
```

Résultat :

Overwriting ILM policy is disabled. Set `setup.ilm.overwrite: true` for enabling.

Index setup finished.

Loading dashboards (Kibana must be running and reachable)

Loaded dashboards




Setting up ML using `setup --machine-learning` is going to be removed in 8.0.0. Please use the ML app instead.


See more: <https://www.elastic.co/guide/en/machine-learning/current/index.html>


Loaded machine learning job configurations



Loaded Ingest pipelines

Pour découvrir vos logs, rendez-vous dans <http://localhost:5601/> et sur le menu à gauche cliquez sur Discover:



 Home

 Home



Recently viewed 
No recently viewed items

 **Kibana** 


- Discover
- Dashboard
- Canvas
- Maps
- Machine Learning
- Visualize

 **Observability** 


- Logs
- Metrics
- APM
- Uptime

 **Security** 

- SIEM

Management 

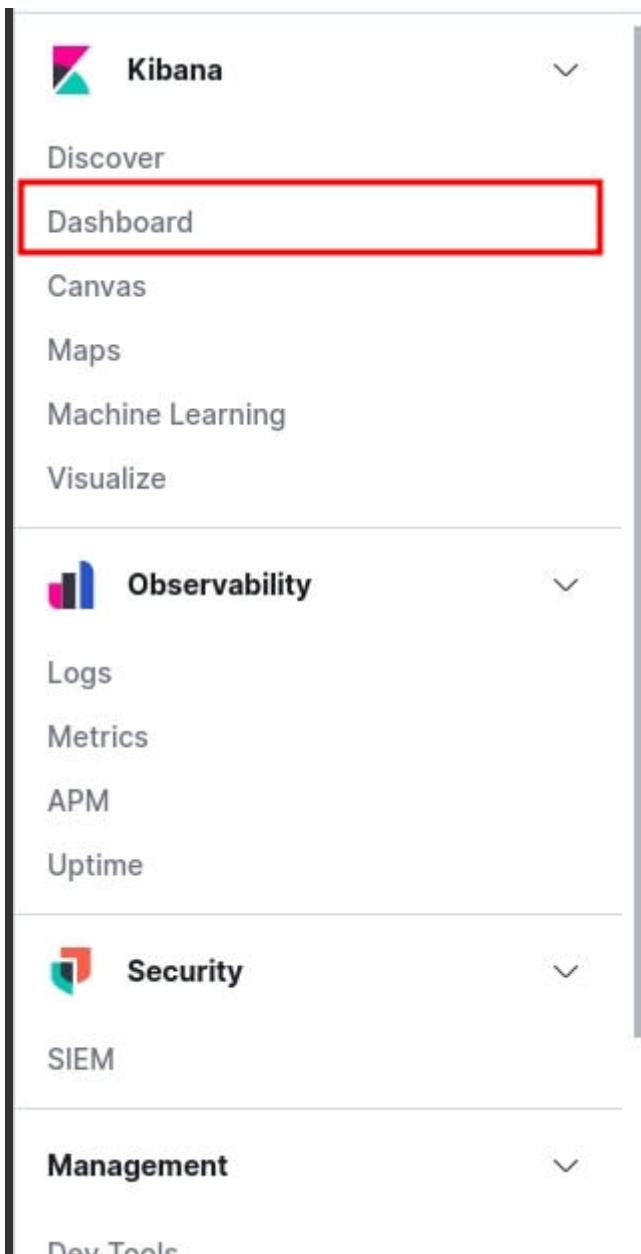
- Dev Tools
- Stack Monitoring
- Stack Management

 Dock navigation

Choisissez ensuite le pattern index **filebeat-*** pour visualiser les logs du module Apache Filebeat:

The screenshot shows the Elastic Discover interface. At the top, the Elastic logo and a search bar are visible. Below the search bar, the 'Discover' tab is selected. The search bar contains the text 'Search'. To the right of the search bar, there are buttons for 'KQL' and a time range selector set to 'Last 15 minutes'. Below the search bar, there is a filter bar with a red box around the text 'filebeat-*'. To the left of the filter bar, there is a search field for field names and a 'Filter by type' dropdown set to '0'. Below the filter bar, there is a list of 'Available fields' with 67 items. The first few fields are: '_id', '_index', '_score', '_type', '@timestamp', 'agent.ephemeral_id', 'agent.hostname', 'agent.id', 'agent.name', and 'agent.type'. To the right of the filter bar, there is a histogram showing '3 hits' for the time range 'Jul 12, 2021 @ 15:10:31.862 - Jul 12, 2021 @ 15:25:31.862'. Below the histogram, there is a table with two columns: 'Time' and 'Document'. The first row of the table shows a timestamp 'Jul 12, 2021 @ 15:24:45.000' and a document snippet containing fields like '@timestamp', 'agent.ephemeral_id', 'agent.hostname', 'agent.id', 'agent.name', and 'agent.type'.

L'étape suivante est de visualiser notre tableau de bord afin de visualiser la collection de visualisations en temps réel issue par notre module Apache Filebeat. Pour ce faire, sur le menu à gauche cliquez sur Dashboard:



Recherchez et cliquez sur les Dashboards Apache:

Dashboards

🔍 apache

<input type="checkbox"/>	Title	Description	Tags
<input type="checkbox"/>	[Filebeat Apache] Access and error logs ECS	Filebeat Apache module dashboard	

Rows per page: 20 ▾

Vous obtenez ainsi le dashboard suivant:



Search

KQL



Last 15 minutes

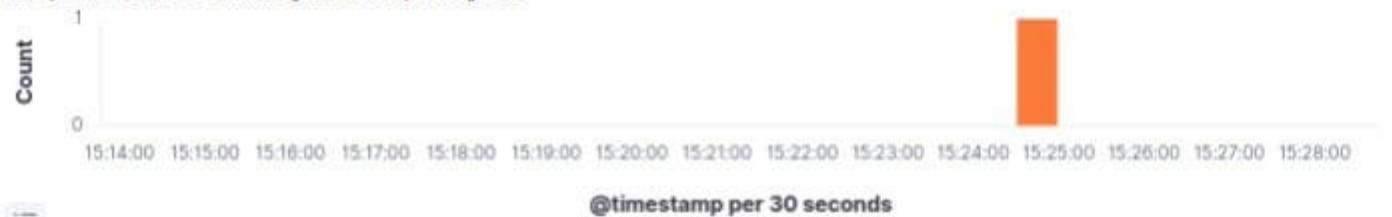


+ Add filter

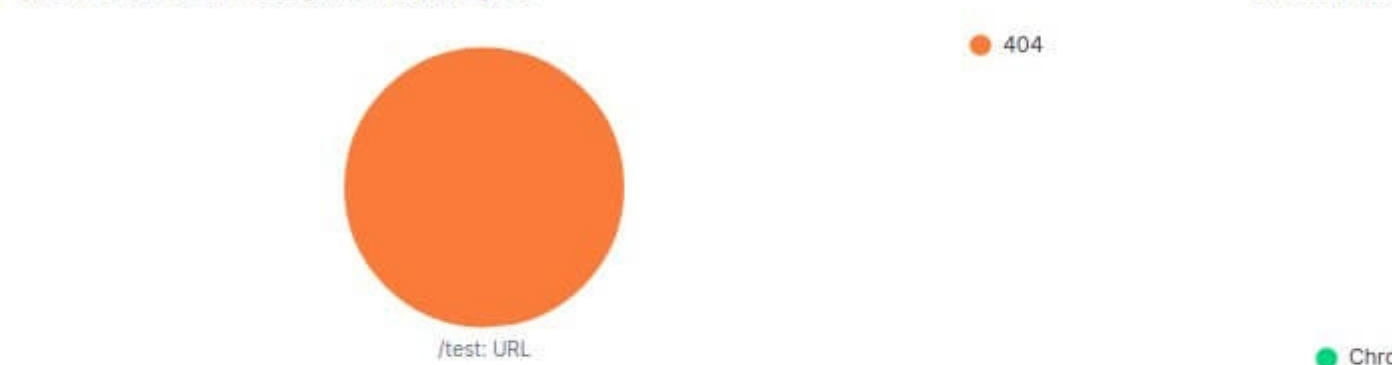
Unique IPs map [Filebeat Apache] ECS



Response codes over time [Filebeat Apache] ECS



Top URLs by response code [Filebeat Apache] ECS



Error logs over time [Filebeat Apache] ECS



No results found

Apache errors log [Filebeat Apache] ECS

Dans notre beau dashboard, nous observons:

- Une carte monde qui géolocalise la provenance des requêtes selon les adresses IP sources
- Un graphe du taux de code de réponse HTTP dans le temps
- Des graphiques circulaire avec des statistiques sur les principales URL par code de réponse HTTP
- Un autre sur la répartition des navigateurs
- Un graphe sur les logs d'erreurs au fil du temps
- Puis en dessous comme pour le tableau de bord des logs système, on a les trames des logs que l'on peut dérouler pour avoir plus d'informations, triables par date, par machine, user-agent, ip ou n'importe quels champs.

Les tableaux de bord restent personnalisables, on peut ajouter, supprimer et réorganiser les fonctions des dashboards à notre guise.

Combiner Filebeat et Logstash

Configuration de Filebeat pour envoyer les logs à Logstash

Si vous souhaitez utiliser Logstash pour effectuer un traitement supplémentaire sur les données collectées par Filebeat, vous devez **configurer Filebeat pour utiliser Logstash**.

Avant de créer le pipeline Logstash, vous configurerez Filebeat pour envoyer des lignes de logs à Logstash. Pour cela, modifiez le fichier `/etc/filebeat/filebeat.yml`:

```
- type: log

# Mettre à true pour activer cette l'entrée Filebeat dans Logstash

enabled: true

# Chemins qui doivent être explorés et récupérés

paths:

  - /var/log/*.log

# On ne souhaite plus envoyer directement nos données à elasticsearch (à commenter
comme ci-dessous)
```

```
#output.elasticsearch:
```

```
#hosts: ["localhost:9200"]
```

On souhaite à la place envoyer directement nos données à Logstash (à décommenter comme ci-dessous)

```
output.logstash:
```

```
hosts: ["localhost:5044"]
```

Dans ce fichier de configuration, Filebeat enverra tous les logs à l'intérieur du dossier */var/log/* vers Logstash. Pour prendre en considération notre configuration, on va redémarrer le service Filebeat:

```
sudo systemctl restart filebeat
```

Ensuite créons un fichier *apache.conf* dans le dossier de configuration Logstash situé dans le dossier */etc/logstash/conf.d/* et rajoutons-y les trois sections de configuration principales:

```
input {
```

```
  beats {
```

```
    port => 5044
```

```
  }
```

```
}
```

```
filter {
```

```
  grok {
```

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

```
  }
```

```
  date {
```

```

        match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]

    }

    mutate {

        convert => {

            "response" => "integer"

            "bytes" => "integer"

        }

    }

}

output {

    stdout { codec => rubydebug }

}
}

```

En plugin d'entrée nous utilisons Beats qui écoute par défaut sur le port 5044. Pour le plugin de filtre et de sortie nous utilisons ceux déjà expliqués dans le [chapitre précédent](#).

Vérifions déjà si notre configuration Logstash est correcte. Pour cela il faut commencer par stopper le service Logstash:

```
sudo systemctl stop logstash
```

Lançons ensuite notre analyseur de configuration Logstash:

```
sudo /usr/share/logstash/bin/logstash --path.settings /etc/logstash -f
/etc/logstash/conf.d/apache.conf -t
```

Résultat :

...

Configuration OK

```
[2021-07-12T19:58:12,834][INFO ][logstash.runner] ... Config Validation Result: OK.  
Exiting Logstash
```

Une fois la syntaxe validée, relançons la commande sans l'option `-t` mais avec l'option `--debug`:

```
sudo /usr/share/logstash/bin/logstash --debug --path.settings /etc/logstash -f  
/etc/logstash/conf.d/apache.confCopier
```

Visitons ensuite depuis notre navigateur la page d'accueil d'Apache <http://localhost/> et revenons sur la sortie standard de notre Logstash pour observer le résultat suivant:

```
{  
  
  "timestamp" => "12/Jul/2021:20:03:10 +0200",  
  
  "@timestamp" => 2021-07-20T15:03:10.000Z,  
  
  "tags" => [  
  
    [0] "beats_input_codec_plain_applied"  
  
  ],  
  
  ....  
  
  "@version" => "1",  
  
  "httpversion" => "1.1",  
  
  ...  
  
},  
  
  "event" => {  
  
    "dataset" => "apache.access",  
  
    "module" => "apache"  
  
  },  
  
  "ecs" => {
```



```
    "version" => "1.9.0"
  },
  "fileset" => {
    "name" => "access"
  },
  "clientip" => "::1",
  "ident" => "-",
  "service" => {
    "type" => "apache"
  },
  "request" => "/",
  "response" => 200,
  "input" => {
    "type" => "log"
  },
  "verb" => "GET",
  "auth" => "-",
  "referrer" => "\"-\""
}
```

Copier

Le résultat nous dévoile que nous récupérons les bonnes données. Maintenant, modifions notre fichier *apache.conf* pour communiquer désormais avec elasticsearch:

```
input {
```

```
beats {

    port => 5044

}

}

filter {

    grok {

        match => { "message" => "%{COMBINEDAPACHELOG}" }

    }

    date {

        match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]

    }

    mutate {

        convert => {

            "response" => "integer"

            "bytes" => "integer"

        }

    }

}

output {
```

```
elasticsearch {  
  
    hosts => "localhost:9200"  
  
    index => "apache-%{+YYYY.MM.dd}"  
  
}  
}Copier
```

On démarre par la suite le service Logstash avec la commande suivante:

```
sudo systemctl start logstashCopier
```

Après démarrage de notre service, Logstash va lancer automatiquement notre pipeline. Ensuite nous pouvons vérifier si l'index s'est correctement créé, pour cela appelons l'API REST Elasticsearch fournie par elasticsearch, comme suit :

```
curl "localhost:9200/_cat/indices?v"Copier
```

Résultat :

```
yellow open    apache-2021.07.12
```

visualisation dans Kibana

Rendez-vous ensuite sur Kibana depuis l'url <http://localhost:5601/> et allons sur le menu à gauche et cliquons sur "Stack Management" pour visualiser notre index:



Home



Home

Recently viewed



No recently viewed items



Kibana



Discover

Dashboard

Canvas

Maps

Machine Learning

Visualize



Observability



Logs

Metrics

APM

Uptime



Security



SIEM

Management



Dev Tools

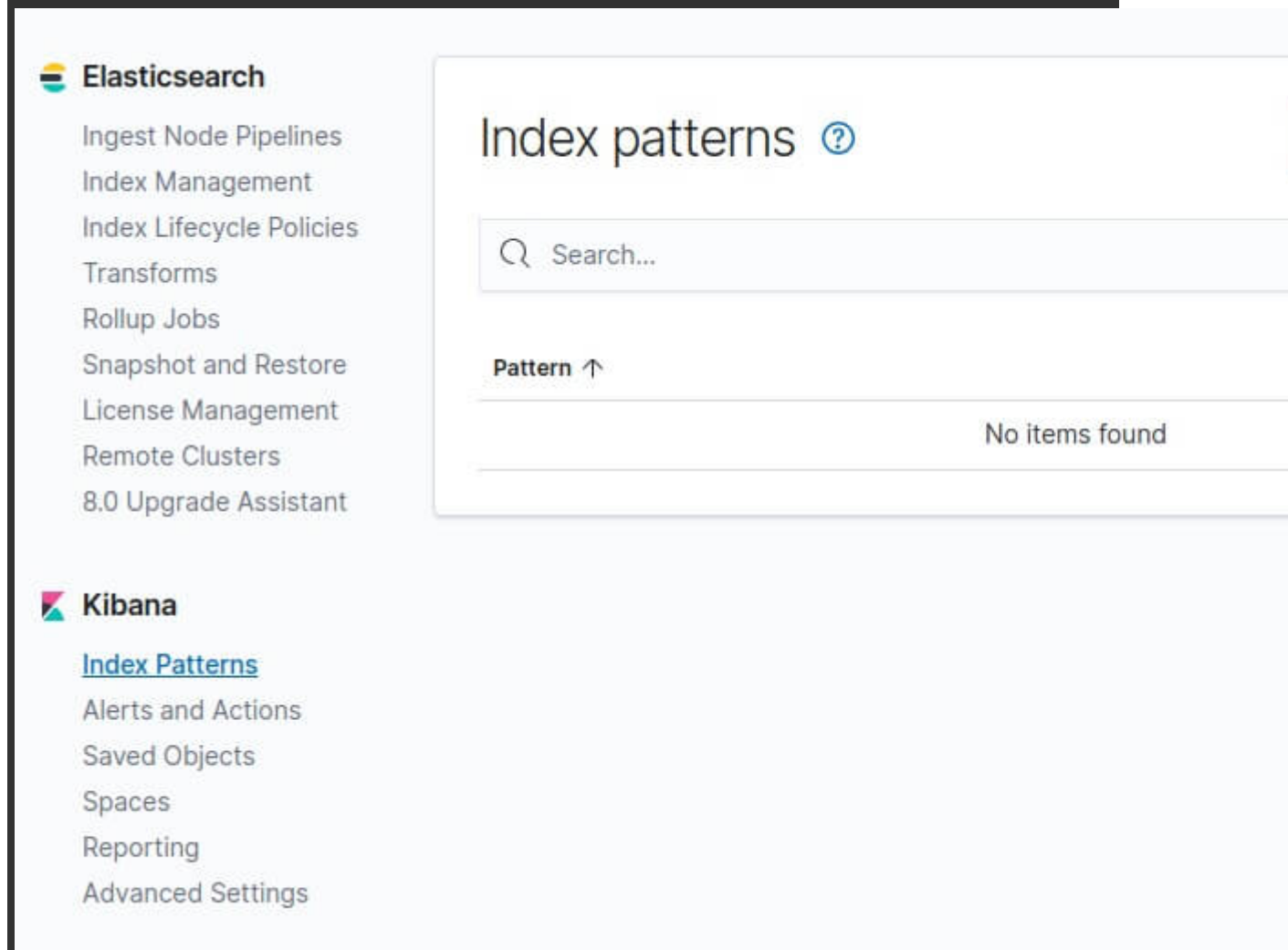
Stack Monitoring

Stack Management



Dock navigation

Comme pour le chapitre précédent, nous allons ajouter un nouveau pattern index dans kibana afin de prendre en considération nos index quotidiens Apache récupérés par Elasticsearch. Pour ce faire, cliquons sur "Index Patterns" sur le volet à gauche et cliquons sur le bouton "Create index pattern" :



Dans notre cas le préfix de nos indexs apache est "apache-", donc le pattern index à créer dans kibana sera **apache-***:

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

☐ Include system indices

Step 1 of 2: Define index pattern

Index pattern

apache-*

You can use a * as a wildcard in your index pattern.
You can't use spaces or the characters \, /, ?, ", <, >, |.

> Next step

✓ **Success!** Your index pattern matches **1 index**.

apache-2020.07.01

Rows per page: 10 ▾

Cliquons ensuite sur "Next Step". Ensuite il nous demande par quel moyen il doit gérer le filtre temporel (timestamp) afin d'affiner nos données par plage horaire. Nous sélectionnerons le champ **@timestamp** :

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

☐ Include system indices

Step 2 of 2: Configure settings

You've defined **apache-*** as your index pattern. Now you can specify some settings before we create it.

Time Filter field name [Refresh](#)

@timestamp

The Time Filter will use this field to filter your data by time. You can choose not to have a time field, but you will not be able to narrow down your data by a time range.

[Show advanced options](#)

[Back](#)

Create index pattern

Enfin, cliquons sur le bouton "Create index pattern" et nous verrons apparaître tous nos champs:

★ apache-*



Time Filter field name: '@timestamp'

Default

This page lists every field in the **apache-*** index and the field's associated core type as recorded by Elasticsearch. To change a field type, use the Elasticsearch [Mapping API](#)

Fields (36)

Scripted fields (0)

Source filters (0)

Search




All field types ▾


Name	Type	Format	Search...	Aggreg...	Excluded
@timestamp 🕒	date		●	●	
@version	string		●		
@version.keyword	string		●	●	
_id	string		●	●	
_index	string		●	●	
_score	number				
_source	_source				
_type	string		●	●	
agent	string		●		
agent.keyword	string		●	●	

Rows per page: 10 ▾

< **1** 2 3 4 >


Pour découvrir nos logs, il suffit d'aller sur le menu à gauche et de cliquer sur "Discover":

 Home

 Home

Recently viewed

No recently viewed items

 **Kibana**

Discover


Dashboard

Canvas

Maps

Machine Learning

Visualize


 **Observability**

Logs

Metrics

APM

Uptime

 **Security**


SIEM

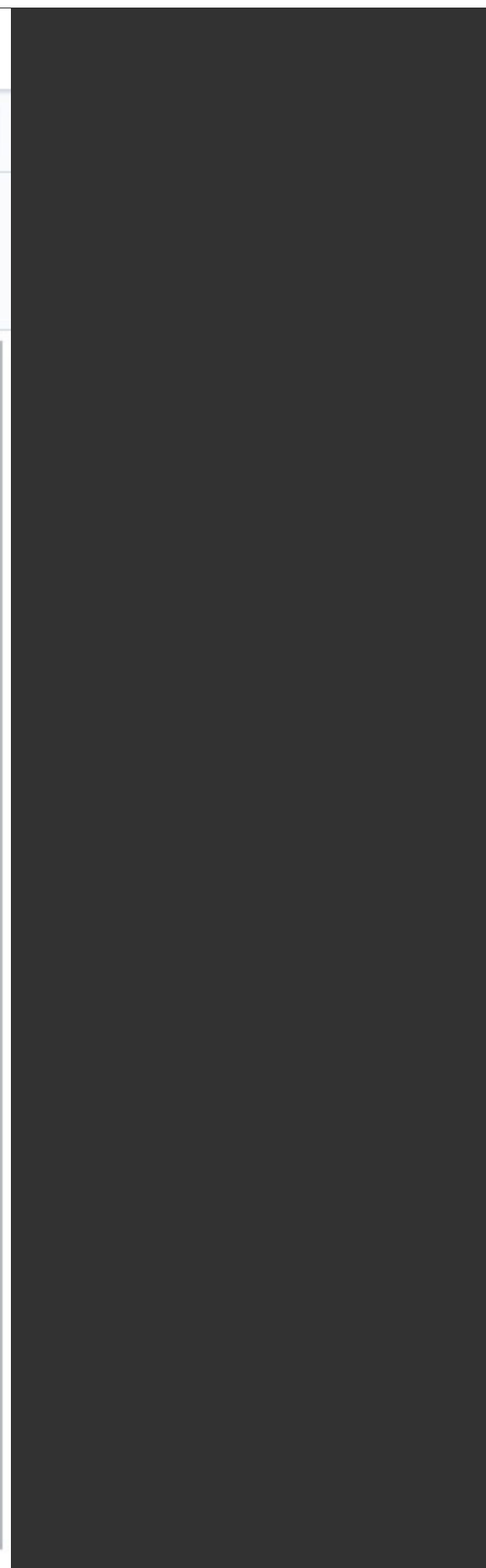
Management

Dev Tools

Stack Monitoring

Stack Management

 Dock navigation



Ensuite, faisons quelques visites depuis notre navigateur sur la page d'accueil d'Apache <http://localhost/> et revenons sur la page de Discover en choisissant le bon pattern index `apache-*` :

The screenshot shows the Elastic Discover interface. At the top, the Elastic logo and a search bar are visible. Below the navigation bar, the 'Discover' tab is selected. A search bar contains the text 'Search'. Below the search bar, a filter is applied, highlighted with a red box, showing 'apache-*'. To the right of the filter, a bar chart displays the distribution of hits over time. The chart shows two bars: one at 18:30:00 with a count of 1, and another at 18:35:00 with a count of 2. Below the chart, a table lists the hits. The table has two columns: 'Time' and 'Document'. The first row shows a hit at 'Jul 12, 2021 @ 18:45:10.419' with a document containing fields like '@timestamp', 'agent.ephemeral_id', 'agent.hostname.keyword', 'agent.id.keyword', and 'agent.type'.

elastic

Search Elastic

Discover

Search

+ Add filter

apache-*

Search field names

Filter by type 0

Available fields 32

- _id
- _index
- _score
- _type
- @timestamp
- @version
- agent.ephemeral_id
- agent.hostname
- agent.id
- agent.name

8 hits

Jul 12, 2021 @ 18:30:15.7

Count

Time

Document

Time	Document
> Jul 12, 2021 @ 18:45:10.419	@timestamp: Jul 12, 2021 @ 18:45:10.419 a264d12f164d agent.ephemeral_id agent.hostname.keyword: G718 agent.id.keyword: 963f32df-c ThinkPad-T14-Gen-1 agent.type

Conclusion

Une consommation de ressources a fait de Logstash un maillon faible de la pile ELK. Pourtant, malgré ces défauts, Logstash reste un composant crucial de la pile, et ce principalement pour manipuler les données avant de les envoyer à Elasticsearch.

Elastic a fait de grands pas en essayant d'atténuer ces douleurs en introduisant Beats, ce qui a permis aux utilisateurs de créer et de configurer plusieurs pipelines de journalisation résilientes et finalement de **faire la journalisation avec ELK beaucoup plus fiable.**

Comprendre et utiliser Metricbeat dans la stack ELK

Dans ce chapitre, nous allons apprendre à utiliser Metricbeat dans la suite ELK en analysant les systèmes et Docker

Introduction

Nous avons vu que la pile ELK, traditionnellement composée d'Elasticsearch, Logstash et Kibana, comprend désormais également un quatrième élément nommé Beats qui comprend une variété d'expéditeurs de données différents.

Dans le [chapitre précédent sur Filebeat](#), nous avons décrit comment utiliser Filebeat pour envoyer des fichiers logs dans la pile. Dans ce chapitre, nous allons **apprendre à utiliser Metricbeat** qui fait partie de la famille d'expéditeurs Beats les plus populaires.

Qu'est-ce que Metricbeat ?

À l'heure actuelle des systèmes et des environnements informatiques très complexes, la surveillance des métriques du système est devenue importante car elle permet d'**augmenter la disponibilité et la fiabilité du système** et permet entre autres aux équipes informatiques de réagir rapidement à toute panne si elle se produit. Différents outils existent pour aider les équipes à récupérer ce type de données dont Metricbeat.

Metricbeat a évolué à partir de [Topbeat](#) et, comme les autres Beats, il est basé sur le framework Go nommé [Libbeat](#). C'est un agent léger qui peut être installé sur les serveurs cibles pour collecter périodiquement des métriques à partir de vos serveurs cibles. Il peut s'agir de métriques de système d'exploitation telles que le processeur ou la mémoire ou des données liées aux services exécutés sur le serveur. C'est un expéditeur léger qui peut être

installé sur les serveurs cibles pour **collecter et expédier périodiquement diverses métriques de système et de service** vers une destination de sortie spécifiée.

C'est donc un agent qui est installé pour **mesurer les performances de vos serveurs**, ainsi que celles des différents services externes qui s'exécutent sur eux. Par exemple, vous pouvez utiliser Metricbeat pour surveiller à la fois la consommation du processeur/mémoire de votre serveur et à la fois les métriques de performances de vos conteneurs.

Tout comme pour Filebeat, il peut être configuré pour envoyer directement la sortie à Elasticsearch ou à Logstash si vous souhaitez transformer les données au préalable.

Installation et configuration de Metricbeat

Installation de Metricbeat

Avant de commencer l'installation, assurez-vous d'avoir installé Elasticsearch pour stocker et rechercher nos données, et d'avoir installé Kibana pour les visualiser et les gérer (mon tuto d'installation est disponible [ici](#)).

Comme pour Filebeat, il existe plusieurs façons d'installer Metricbeat. Dans notre cas nous allons comme pour Filebeat, soit **installer Metricbeat depuis le gestionnaire de paquets par défaut** de notre distribution. Pour ce faire, vous devrez d'abord mettre à niveau votre système et vos paquets:

```
sudo apt update -y && sudo apt upgrade -yCopier
```

Pour les machines appartenant à la famille debian, vous devrez peut-être installer le paquet `apt-transport-https` avant de continuer:

```
sudo apt-get install apt-transport-httpsCopier
```

Téléchargez et installez ensuite la clé de signature publique (étape non obligatoire, si vous avez suivi le chapitre précédent) :

Sous la famille debian:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo  
apt-key add -Copier
```

Sous la famille redhat:

```
sudo rpm --import https://packages.elastic.co/GPG-KEY-  
elasticsearchCopier
```

L'étape suivante consiste à ajouter le dépôt Elastic sur votre système (étape non obligatoire, si vous avez suivie le chapitre précédent) :

Sous la famille debian:

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main"  
| sudo tee -a /etc/apt/sources.list.d/elastic-7.x.listCopier
```

Sous la famille redhat, créez un fichier et nommez le par exemple dans le répertoire */etc/yum.repos.d/*, contenant:

```
[elastic-7.x]  
  
name=Elastic repository for 7.x packages  
  
baseurl=https://artifacts.elastic.co/packages/7.x/yum  
  
gpgcheck=1  
  
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch  
  
enabled=1  
  
autorefresh=1  
  
type=rpm-mdCopier
```

Il ne vous reste plus qu'à mettre à jour vos référentiels et d'installer Metricbeat:

Sous la famille debian:

```
sudo apt-get update && sudo apt-get install metricbeatCopier
```

Sous la famille redhat:

```
sudo yum install metricbeatCopier
```

Pour exécuter Metricbeat, utilisez la commande suivante:

```
sudo systemctl start metricbeatCopier
```

Si jamais vous rencontrez des problèmes d'initialisation, veuillez vérifier les logs du service Metricbeat à l'aide de la commande suivante :

```
sudo journalctl -f -u metricbeatCopier
```

Configuration de Metricbeat

Le fichier de configuration de Metricbeat se retrouve dans `/etc/metricbeat/metricbeat.yml`. Dans ce fichier, assurez-vous bien que la configuration Metricbeat possède les bonnes informations pour communiquer avec Kibana et Elasticsearch, si besoin décommentez les lignes suivantes :

```
output.elasticsearch:
```

```
  hosts: ["localhost:9200"]
```

```
  username: "" #(si pas de login/mot de passe ne rien mettre)
```

```
  password: "" #(si pas de login/mot de passe ne rien mettre)
```

```
...
```

```
setup.kibana:
```

```
  host: "localhost:5601"
```

```
...
```

```
# Optionnelle

metricbeat.config.modules:

  path: ${path.config}/modules.d/*.yaml

  reload.enabled: true

  reload.period: 15sCopier
```

Ci-dessous une explication détaillée sur les options de configuration Metricbeat utilisées:

- **metricbeat.config.modules** : où nous retrouverons l'option **path** qui est l'emplacement des fichiers de configuration des modules Metricbeat situés par défaut dans le dossier `/etc/metricbeat/modules.d/`, et pour les cibler par défaut Metricbeat utilise la regex `*.yaml`. Par ailleurs, nous avons également la possibilité d'activer ou non le rechargement automatique de la configuration qui est par défaut à **false** dans l'option **reload.enabled**. Si vous passez la valeur **true** comme moi, alors Metricbeat surveillera périodiquement (ici toutes les 15 secondes) vos fichiers de configuration et si des changements sont détectés, il rechargera l'ensemble de la configuration.
- **setup.kibana** : pour que les tableaux de bord fonctionnent, nous devons spécifier le point de terminaison Kibana. Vous devrez entrer l'URL de votre hôte Kibana et vos informations d'identification (nom d'utilisateur/mot de passe) si nécessaire.
- **output.elasticsearch** : spécifie la sortie à laquelle nous envoyons les métriques Metricbeat. Nous utilisons Elasticsearch, vous devrez donc fournir l'hôte, le protocole et les informations d'identification Elasticsearch si nécessaire.

Pour initialiser le service à chaque démarrage de la machine, lancez la commande suivante:

```
sudo systemctl enable metricbeatCopier
```

Les modules Metricbeat

Comme pour Filebeat, il existe une multitude de modules qu'on peut utiliser sur l'expéditeur Metricbeat qui contiennent des définitions de **collecte et de connexion spécifiques à un service**. Ils définissent les métriques spécifiques à collecter et expédier, la fréquence à laquelle les collecter et comment se connecter auprès du service concerné.

Metricbeat prend en charge un nombre croissant de modules pour expédier des métriques, telle que ceux d'Apache, Système, MySQL, Docker etc ... Vous retrouverez l'intégralité des modules dans cette [page](#).

Collecter et expédier les métriques système d'un serveur

Dans cet exemple, nous allons configurer Metricbeat pour utiliser le [module system](#) afin de surveiller et collecter les métriques systèmes du serveur, telles que l'utilisation du processeur et de la mémoire, le réseau, le disque etc.

Avant de suivre ces étapes, vérifiez qu'Elasticsearch et Kibana sont en cours d'exécution et qu'Elasticsearch est prêt à recevoir des données de MetricBeat:

```
sudo systemctl status elasticsearch kibanaCopier
```

Résultat :

```
• elasticsearch.service - Elasticsearch

   Loaded: loaded (/lib/systemd/system/elasticsearch.service;
   disabled; vendor preset: enabled)

   Active: active (running) since Tue 2021-07-13 16:58:31 CEST; 1h
   24min ago

   13 16:58:31 G7189-ThinkPad-T14-Gen-1 systemd[1]: Started
   Elasticsearch.

• kibana.service - Kibana

   Loaded: loaded (/etc/systemd/system/kibana.service; disabled;
   vendor preset: enabled)

   Active: active (running) since Tue 2021-07-13 16:58:16 CEST; 1h
   24min ago
```

Par défaut plusieurs métriques sont activées. Pour désactiver un ensemble de métriques par défaut, mettez-les en commentaire dans le fichier de configuration :

```
- module: system

  period: 10s
```



```
metricsets:

  - cpu

  - load

  - memory

  - network

  - process

  #- process_summary

  #- socket_summary

  #- entropy

  #- core

  #- diskio

  #- socket

  #- service

  #- users

process.include_top_n:

  by_cpu: 10      # include top 10 processes by CPU

  by_memory: 10   # include top 10 processes by memory

...Copier
```

Dans mon cas je récupère périodiquement toutes les 10 secondes des métriques de mon serveur sur le cpu, mémoire, réseaux, disques et processus en cours d'exécution ainsi que le top 10 (au lieu de 5 par défaut) des processus qui consomment le plus de cpu et de mémoire.

Pour configurer et exécuter le module system, lancez la commande suivante:

```
sudo metricbeat modules enable systemCopier
```

Pour voir la liste des modules activés et désactivés, exécutez :

```
sudo metricbeat modules listCopier
```

Résultat :

```
Enabled:
```

```
system
```

```
Disabled:
```

```
activemq
```

```
aerospike
```

```
apache
```

```
...
```

Ensuite lancez la sous-commande `setup` pour charger les tableaux de bord dans Kibana (la commande peut prendre un peu de temps pour se terminer):

```
sudo metricbeat setupCopier
```

Résultat :

```
Overwriting ILM policy is disabled. Set `setup.ilm.overwrite: true`  
for enabling.
```

```
Index setup finished.
```

```
Loading dashboards (Kibana must be running and reachable)
```

```
Loaded dashboards
```

Comme pour Filebeat pour découvrir vos logs, rendez-vous dans <http://localhost:5601/> et sur le menu à gauche cliquez sur Discover:



Home



Home

Recently viewed



No recently viewed items



Kibana



Discover

Dashboard

Canvas

Maps

Machine Learning

Visualize



Observability



Logs

Metrics

APM

Uptime



Security



SIEM

Management



Dev Tools

Stack Monitoring

Stack Management



Dock navigation

Choisissez ensuite le pattern index **metricbeat-*** pour visualiser les logs du module system Metricbeat:

The screenshot shows the Elastic Discover interface. At the top, the Elastic logo and a search bar are visible. Below the search bar, the 'Discover' tab is selected. A filter box on the left contains the pattern 'metricbeat-*', which is highlighted with a red rectangle. Below this, a list of available fields is shown, including '_id', '_index', '_score', '_type', '@timestamp', 'agent.ephemeral_id', 'agent.hostname', 'agent.id', 'agent.name', 'agent.type', 'agent.version', 'ecs.version', 'event.dataset', 'event.duration', 'event.module', 'host.architecture', 'host.containerized', 'host.cpu.pct', and 'host.hostname'. On the right, a bar chart displays the count of hits over time, with a peak around 18:29:00. Below the chart, a table of hits is shown, with the first three hits visible. Each hit includes a timestamp and a document containing fields like '@timestamp', 'agent.hostname', 'agent.name', 'agent.version', 'ecs.version', and 'event.duration'.

elastic Search Elastic

Discover

Search KQL Last 15 m

+ Add filter

metricbeat-*

Search field names

Filter by type 0

Available fields 240

_id

_index

_score

_type

@timestamp

agent.ephemeral_id

agent.hostname

agent.id

agent.name

agent.type

agent.version

ecs.version

event.dataset

event.duration

event.module

host.architecture

host.containerized

host.cpu.pct

host.hostname

1,813 hits Jul 13, 2021 @ 18:29:12.300 - Jul 13, 2021 @ 18:36:12.300

Count

@timestamp

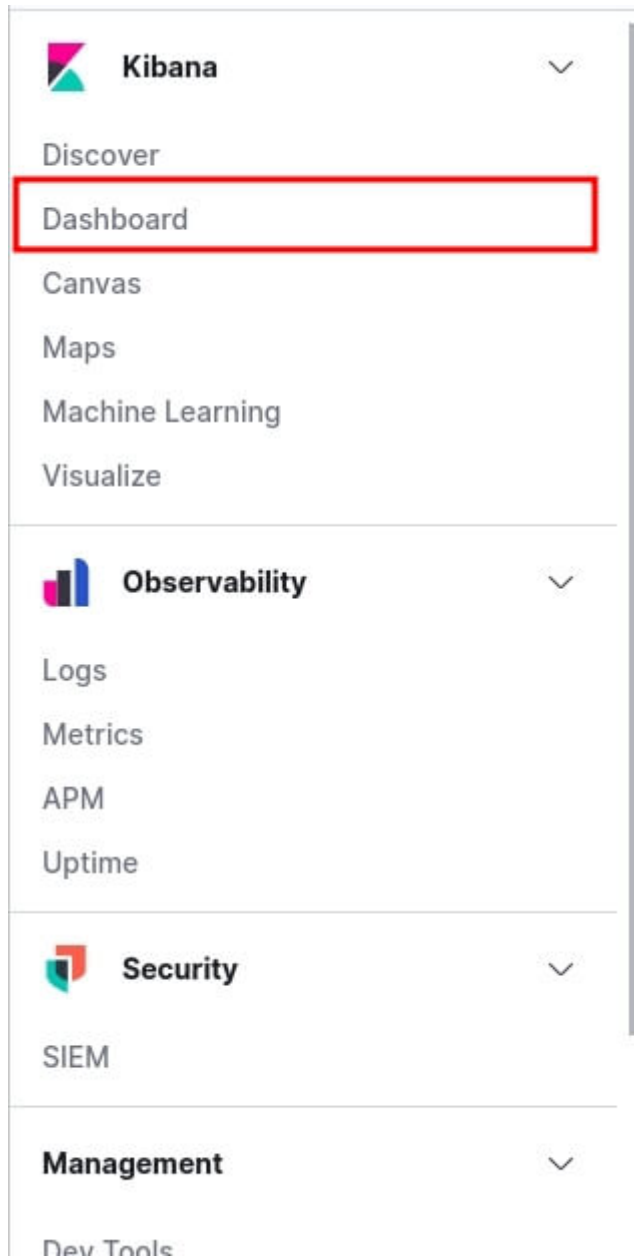
Time Document

> Jul 13, 2021 @ 18:44:06.122 @timestamp: Jul 13, 2021 @ 18:44:06.122 agent.hostname: 1e79793a542c agent.name: G71d7898e40c549 agent.version: 7.13.3 ecs.version: 7.13.3 event.duration: 513.0 event.module: event.type: log event.type: log

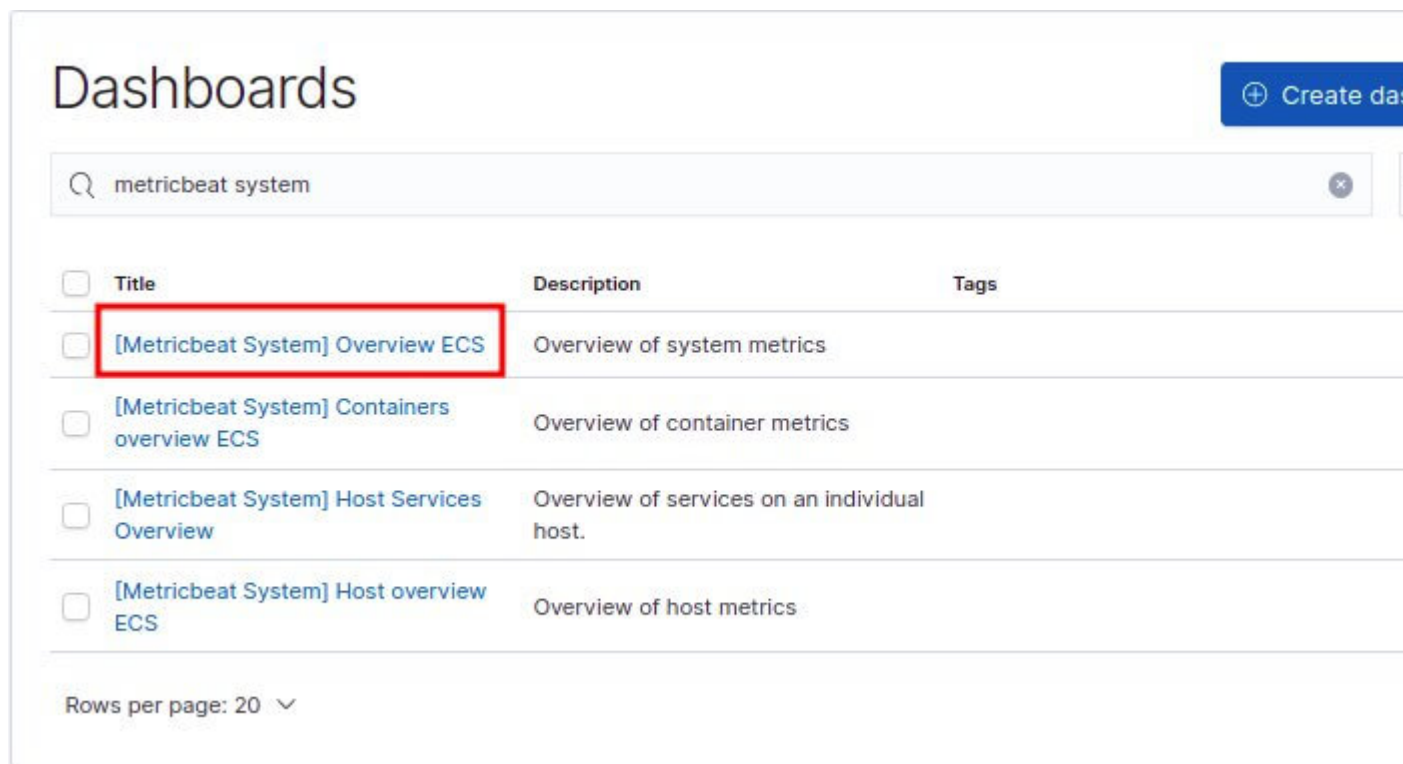
> Jul 13, 2021 @ 18:44:06.122 @timestamp: Jul 13, 2021 @ 18:44:06.122 agent.hostname: 1e79793a542c agent.name: G71d7898e40c549 agent.version: 7.13.3 ecs.version: 7.13.3 event.duration: 513.1 event.module: event.type: log event.type: log

> Jul 13, 2021 @ 18:44:06.122 @timestamp: Jul 13, 2021 @ 18:44:06.122 agent.hostname: 1e79793a542c agent.name: G71d7898e40c549 agent.version: 7.13.3 ecs.version: 7.13.3 event.duration: 513.1 event.module: event.type: log event.type: log

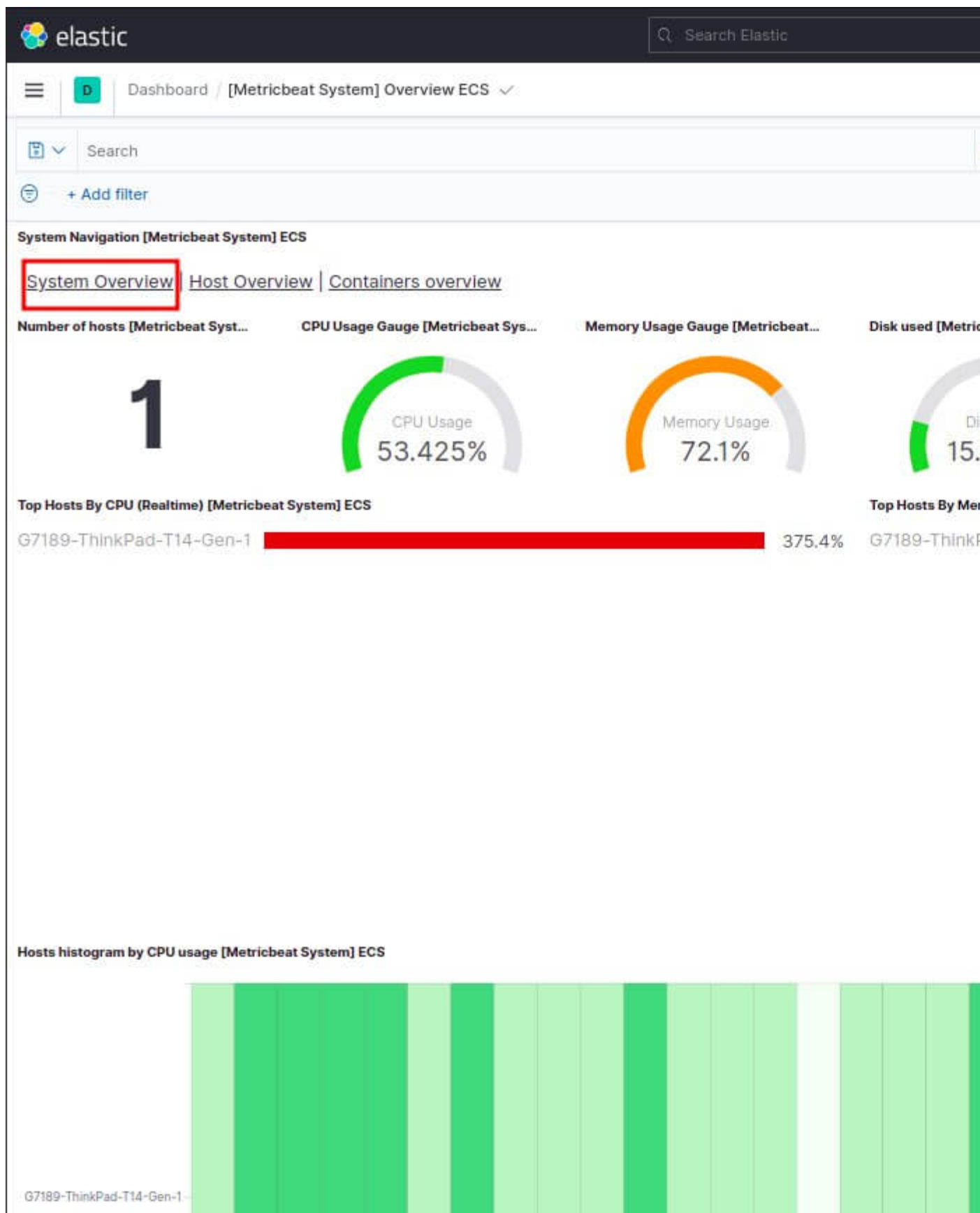
L'étape suivante est de visualiser notre tableau de bord afin de visualiser la collection de visualisations en temps réel issue par notre module system Metricbeat. Pour ce faire, sur le menu à gauche cliquez sur Dashboard:



Recherchez et cliquez sur les Dashboards du module System:



Vous obtenez ainsi 3 types de dashboard. Le premier type nommé "System Overview" ressemble à l'image ci-dessous et vous donne une moyenne la consommation de votre/vos serveur(s) concernant leur cpu, mémoire, réseaux, disque ainsi que le nombre de machines utilisant ce même module system Metricbeat:



Dans le second nommé "Host Overview", vous avez avec les mêmes informations sur les ressources consommées que le premier dashboard mais en plus détaillées, ainsi que des

informations sur les processus comme par exemple le top 10 des processus qui consomment le plus de cpu/mémoire que j'ai configuré plus haut dans le fichier de Configuration du module system Metricbeat:

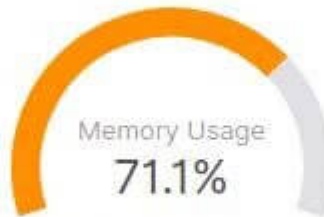
System Navigation [Metricbeat System] ECS

[System Overview](#) [Host Overview](#) [Containers overview](#)

CPU Usage Gauge [Metricbeat System] ...



Memory Usage Gauge [Metricbeat Syst...



Load Gauge [Metricbeat System] ECS



Swap usage [Metricbeat System] ECS



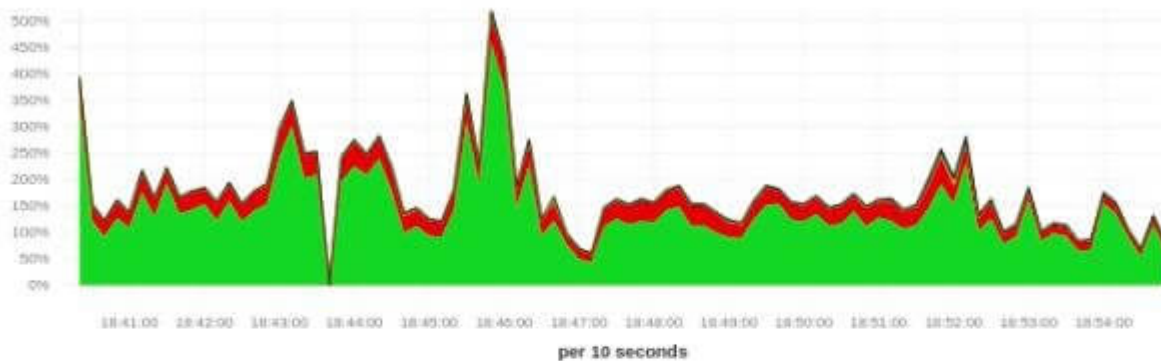
Memory usage vs total [Metricbeat Syst...

Memory usage
10.9GB
Total Memory 15.3GB

Number of processes [Metricbeat Syste...

45
Processes

CPU Usage [Metricbeat System] ECS



Processes By Memory [Metricbeat System] ECS

kolourpaint	-
node	-
java	10.2%
teams	3.65%
gnome-software	3.4%
code	2.525%
mysqld	2.5%
chrome	2.12%
cinnamon	1.4%
metricbeat	1%

Interfaces by Incoming traffic [Metricbeat System] ECS

Dans le dernier nommé "Containers overview", vous avez des informations sur les services qui tournent sur votre machine ainsi que les ressources qu'ils consomment:

System Navigation [Metricbeat System] ECS

[System Overview](#) | [Host Overview](#) | [Containers overview](#)

Container CPU usage [Metricbeat System] ECS

 Export

Container ID	Process name	CPU user
elasticsearch.service	java	530,520,000,000
kibana.service	node	212,110,000,000
mysql.service	mysqld	60,470,000,000
metricbeat.service	metricbeat	21,760,000,000

Container Memory stats [Metricbeat System] ECS

 Export

Container ID	Process name	Usage	Max usage	Page faults	Pages in memory	Pages out of memory	Inactive
kibana.service	node	506.9MB	757.1MB	1,082,292.484	1,141,329.484	1,019,129.677	74.9MB
elasticsearch.s...	java	1.7GB	1.7GB	624,231.375	810,297	370,538.091	68.5MB
mysql.service	mysqld	406.3MB	445.4MB	105,369	128,436	26,047	23.9MB
metricbeat.ser...	metricbeat	76.1MB	76.6MB	20,959	18,100.5	0	4.3MB

Container Block IO [Metricbeat System] ECS

 Export

Container ID	Process name
kibana.service	node
elasticsearch.service	java
mysql.service	mysqld
init.scope	systemd
metricbeat.service	metricbeat

Collecter et expédier les métriques d'un service (Docker)

Dans la partie précédente, nous avons expliqué comment utiliser Metricbeat pour envoyer des métriques systèmes. Il est maintenant temps de collecter et expédier des métriques pour un service ciblé. Dans cet exemple, je vais **surveiller les métriques à partir des conteneurs Docker** exécutés sur un système hôte. Pour ce faire nous utiliserons le afin d'avoir des

données prédéfinies pour collecter et envoyer des métriques et des statistiques de service de conteneur Docker à Logstash ou Elasticsearch.

Pour récupérer les métriques des conteneurs Docker, nous allons utiliser le [module docker Metricbeat](#) qui est livré avec un certain nombre d'ensemble de métriques par défaut dont nous avons besoin, tels que les informations sur les conteneurs ainsi que leur consommation du cpu, mémoire, disque, network etc...

Tout d'abord, je vais supposer que vous disposez déjà d'un environnement Docker fonctionnel sur votre système, si ce n'est pas le cas merci de suivre mon [article sur l'installation de docker](#). Vous aurez besoin d'au moins un conteneur en cours d'exécution dans Docker dans le but d'envoyer des données utiles à Elasticsearch et Kibana. Pour cet article, nous allons exécuter deux conteneurs, qui sont:

```
docker run -d -p 9000:80 --name nginx-metricbeat nginxCopier
```

et

```
docker run -d -p 9001:80 --name httpd-metricbeat httpdCopier
```

On s'assure ensuite que nos conteneurs s'exécutent correctement:

```
docker psCopier
```

Résultat :

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS		NAMES
41724bbff68d	httpd	"httpd-foreground"	7 seconds ago
Up 6 seconds	0.0.0.0:9001->80/tcp, :::9001->80/tcp		httpd-metricbeat
83f78bc139ab	nginx	"/docker-entrypoint...."	14 seconds ago
Up 13 seconds	0.0.0.0:9000->80/tcp, :::9000->80/tcp		nginx-metricbeat

Ensuite, nous activons manuellement le module Docker:

```
sudo metricbeat modules enable dockerCopier
```

Maintenant que le module est activé, modifions sa configuration. Voici à quoi cela ressemble mon fichier :

```
- module: docker

metricsets:

  - container

  - cpu

  - diskio

  - event

  - healthcheck

  - info

  - memory

  - network

  - image

period: 10s

hosts: ["unix:///var/run/docker.sock"]Copier
```

Il s'agit d'une configuration minimale suffisante pour lancer Metricbeat. Nous avons spécifié 9 ensembles de métriques, y compris l'ensemble de métriques **image** qui est non inclus par défaut.

De plus, le module Docker a besoin d'accéder au démon Docker. Par défaut, Docker écoute sur le socket Unix `unix:///var/run/docker.sock`. Nous pouvons utiliser ce socket pour communiquer avec le démon depuis un conteneur. Grâce à ce point de terminaison, Docker expose son API qui peut être utilisée pour obtenir un flux de tous les événements et statistiques générés par Docker.

Le prochain champ de configuration important que nous devons mentionner est **period** où j'ai laissé la valeur par défaut. Ce champ définit la fréquence à laquelle Metricbeat accède à l'API Docker.

Attention

Selon la documentation officielle de Metricbeat, il est fortement recommandé d'exécuter un module Docker avec une période d'au moins 3 secondes ou plus. C'est parce que la demande à l'API Docker prend elle-même jusqu'à 2 secondes, donc si vous spécifiez moins de 3 secondes, cela peut entraîner des délais d'attente de demande et aucune donnée ne sera renvoyée.

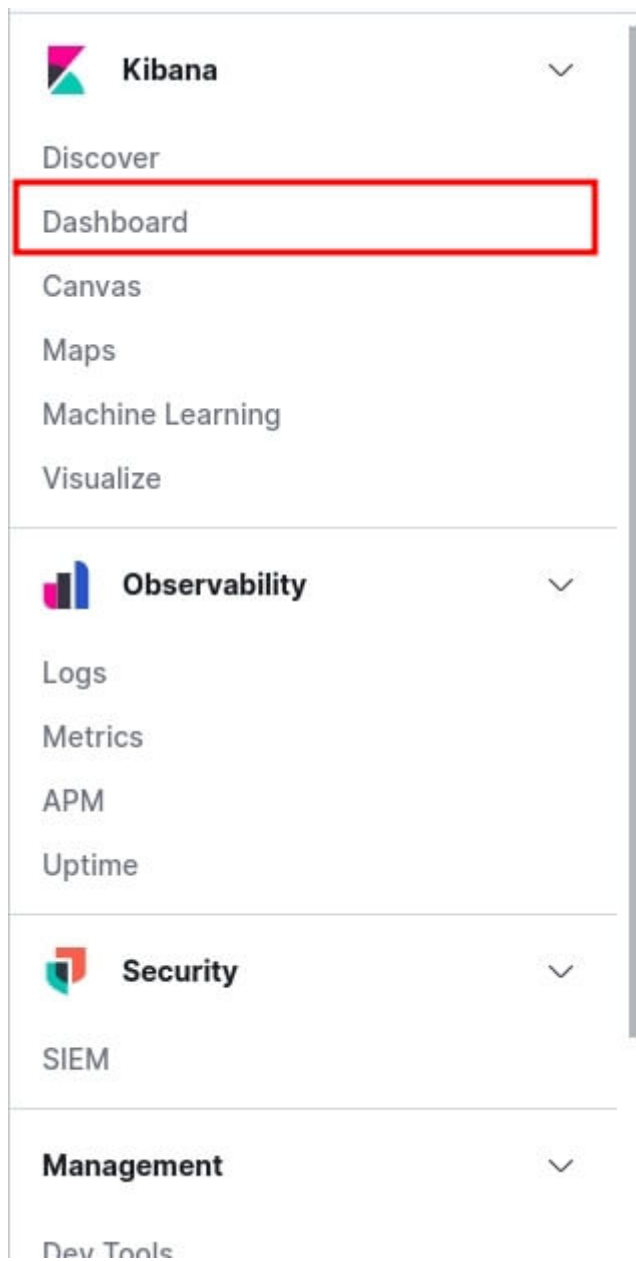
Maintenant, tout est prêt pour exécuter Metricbeat. Un dernier conseil est d'avoir votre instance Metricbeat aussi proche que possible de Docker (de préférence sur le même hôte) pour minimiser la latence du réseau.

```
sudo metricbeat setup
```

Résultat :

```
Overwriting ILM policy is disabled. Set `setup.ilm.overwrite: true`  
for enabling.  
  
Index setup finished.  
  
Loading dashboards (Kibana must be running and reachable)  
  
Loaded dashboards
```

Comme pour l'exemple précédent, rendez-vous dans <http://localhost:5601/> et sur le menu à gauche cliquez sur Dashboard:




Recherchez et cliquez sur les Dashboards du module Docker:

Dashboards

[+ Create dashboard](#)

You have unsaved changes in the following dashboard.

 **[Metricbeat System] Host overview ECS**

[Continue editing](#) [Discard changes](#)

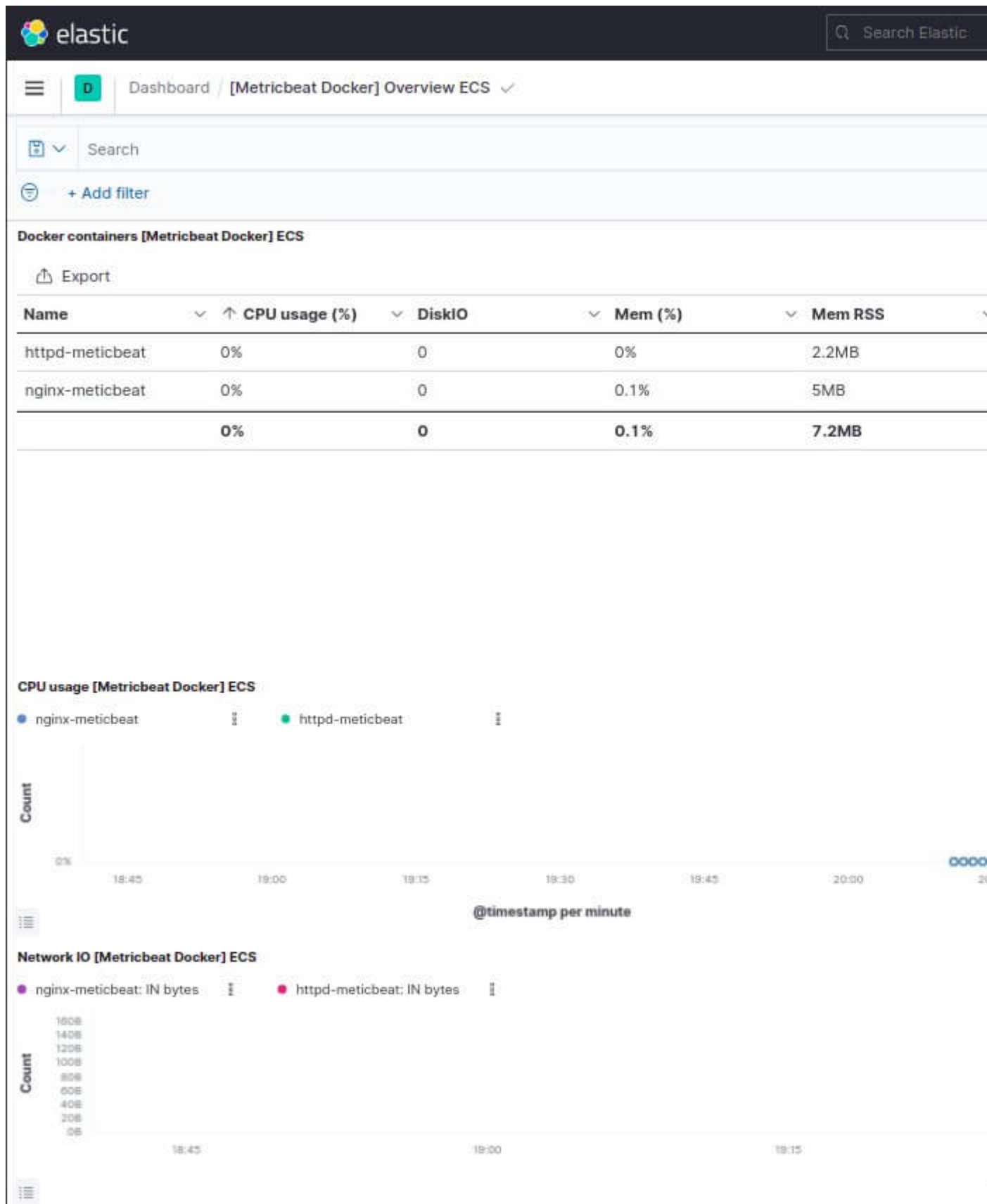
Tags

<input type="checkbox"/>	Title	Description	Tags	Action
<input type="checkbox"/>	<div>[Metricbeat Docker] Overview ECS</div>	Overview of docker containers		

Rows per page: 20 ▼

< 1

Vous obtenez alors le dashboard suivant, avec les statistiques des conteneurs en cours d'exécution ainsi que des données sur leur utilisation de la mémoire et du processeur:



Si jamais vous ne voyez aucune donnée, tentez de restart le service:

```
sudo systemctl restart metricbeat
```

Conclusion

Metricbeat est un expéditeur métrique extrêmement facile à utiliser, efficace et fiable pour surveiller votre système et les processus qui y sont exécutés. Comme les autres expéditeurs de la famille Beats il laisse une faible empreinte sur les ressources et peut être installé et démarré relativement rapidement.

Dans le prochain chapitre nous aborderons l'expéditeur métrique Packetbeat.

[Chapitre précédent](#)

Comprendre et utiliser Packetbeat dans la stack ELK

Dans ce chapitre, nous allons apprendre à utiliser Packetbeat dans la suite ELK en analysant les paquets d'une application Web

Introduction

Nous sommes maintenant confrontés à des entreprises avec des réseaux de plus en plus complexes qui doivent être protégés. De nos jours, il est absolument nécessaire de surveiller votre réseau, chaque organisation doit **collecter toutes les informations sur ses réseaux**. Si vous ne surveillez pas votre réseau, comment pourrez-vous détecter ce qui est un comportement normal ou une attaque ? Il existe des attaques malveillantes évidentes, mais une détection plus précoce est cruciale dans tout incident de cybersécurité. L'un des moyens les meilleurs et les plus pratiques de détecter une activité anormale est **la surveillance du réseau**. Dans cet article, nous allons voir comment est-il possible de surveiller votre réseau avec l'outil Packetbeat, puis d'agréger toutes les informations dans la pile ELK.

Dans le [chapitre précédent sur Metricbeat](#), nous avons décrit comment utiliser Metricbeat pour envoyer des métriques systèmes dans la pile. Dans ce chapitre, nous allons **apprendre à utiliser Packetbeat** qui fait partie de la famille d'expéditeurs Beats les plus populaires.

Qu'est-ce que Packetbeat ?

Packetbeat est un outil de surveillance réseau développé par Elastic qui utilise la bibliothèque [libpcap](#), c'est un expéditeur et un analyseur de données open source pour les paquets réseaux intégrés à la pile ELK (Elasticsearch, Logstash et Kibana) et membre de la famille des expéditeurs Beats (Filebeat, Libbeat, Winlogbeat, etc...). Il fournit des métriques de surveillance réseau en temps réel sur du protocole HTTP, TLS, DNS et de nombreux autres protocoles réseaux.

Ainsi grâce à cet outil de surveillance de paquets de données avec la stack ELK, il peut vous aider à détecter des niveaux inhabituels de trafic réseau et des caractéristiques de paquets inhabituelles, à identifier les sources et les destinations des paquets, à rechercher des chaînes de données spécifiques dans les paquets et à créer un tableau de bord convivial avec des statistiques pertinentes prêtes à l'emploi. Ainsi cela peut contribuer à améliorer vos temps de réponse aux attaques malveillantes.

Dans cet article, nous allons démontrer la plupart des avantages mentionnés ci-dessus. Plus précisément, nous allons **apprendre à utiliser Packetbeat pour surveiller les transactions HTTP d'une application Web** et d'analyser les données en utilisant la stack ELK.

Installation et configuration de Packetbeat

Préparation de l'application web

Afin d'éviter d'installer de nombreux paquets et bibliothèques, nous allons déployer notre application web depuis la technologie Docker. Tout D'abord commencez par télécharger les sources du projet en cliquant [ici](#) et désarchivez ensuite le projet.

Ensuite, je vais supposer que vous disposez déjà d'un environnement Docker fonctionnel sur votre système, si ce n'est pas le cas merci de suivre mon [article sur l'installation de docker](#).

Une fois cette étape réalisée, placez-vous alors à la racine du projet et buildez ensuite votre image avec la commande suivante :

```
docker build -t myapp .Copier
```

Ensuite toujours depuis la racine du projet, lancez la commande suivante pour exécuter vos conteneurs :

```
docker-compose up -dCopier
```

On s'assure ensuite que nos conteneurs s'exécutent correctement sur les bons ports:

```
docker psCopier
```

Résultat :

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS		
NAMES			
97e0bb711609	myapp	"docker-php-entrypoi..."	12 minutes ago
Up 12 minutes	0.0.0.0:80->80/tcp, :::80->80/tcp		
myapp_c			
feae30be6007	mysql:5.7	"docker-entrypoint.s..."	12 minutes ago
Up 12 minutes	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp		
mysql_c			

Ici nous avons un conteneur contenant un serveur web nommé "myapp_c" écoutant sur le port 80 et un autre conteneur mysql écoutant sur le port 3306.

Si tout c'est bien passé, alors visitez la page suivante <http://localhost>, et vous obtiendrez le résultat suivant :

← → ↻ 🏠 🌐 http://localhost

Mon app Packetbeat Accueil Articles

Articles

Nouveau article

Titre *

Nom de l'auteur *

Contenu *

Envoyer

Installation de Packetbeat

Avant de commencer l'installation, assurez-vous d'avoir installé Elasticsearch pour stocker et rechercher nos données, et d'avoir installé Kibana pour les visualiser et les gérer (mon tuto d'installation est disponible [ici](#)).

Comme pour Metricbeat, il existe plusieurs façons d'installer Packetbeat. Dans notre cas nous allons comme pour, soit **installer Packetbeat depuis le gestionnaire de paquets par**

défaut de notre distribution. Pour ce faire, vous devrez d'abord mettre à niveau votre système et vos paquets:

```
sudo apt update -y && sudo apt upgrade -yCopier
```

Pour les machines appartenant à la famille debian, vous devrez peut-être installer le paquet `apt-transport-https` avant de continuer:

```
sudo apt-get install apt-transport-httpsCopier
```

Téléchargez et installez ensuite la clé de signature publique (étape non obligatoire, si vous avez suivie le chapitre précédent) :

Sous la famille debian:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo  
apt-key add -Copier
```

Sous la famille redhat:

```
sudo rpm --import https://packages.elastic.co/GPG-KEY-  
elasticsearchCopier
```

L'étape suivante consiste à ajouter le dépôt Elastic sur votre système (étape non obligatoire, si vous avez suivie le chapitre précédent) :

Sous la famille debian:

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main"  
| sudo tee -a /etc/apt/sources.list.d/elastic-7.x.listCopier
```

Sous la famille redhat, créez un fichier et nommez le par exemple dans le répertoire `/etc/yum.repos.d/`, contenant:

```
[elastic-7.x]  
  
name=Elastic repository for 7.x packages  
  
baseurl=https://artifacts.elastic.co/packages/7.x/yum  
  
gpgcheck=1
```

```
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch  
  
enabled=1  
  
autorefresh=1  
  
type=rpm-mdCopier
```

Il ne vous reste plus qu'à mettre à jour vos référentiels et d'installer Packetbeat:

Sous la famille debian:

```
sudo apt-get update && sudo apt-get install packetbeatCopier
```

Sous la famille redhat:

```
sudo yum install packetbeatCopier
```

Pour exécuter Packetbeat, utilisez la commande suivante:

```
sudo systemctl start packetbeatCopier
```

Si jamais vous rencontrez des problèmes d'initialisation, veuillez vérifier les logs du service Packetbeat à l'aide de la commande suivante :

```
sudo journalctl -f -u packetbeatCopier
```

Configuration de Packetbeat

Le fichier de configuration de Packetbeat se retrouve dans `/etc/packetbeat/packetbeat.yml`. Dans ce fichier, assurez-vous bien que la configuration Packetbeat possède les bonnes informations pour communiquer avec Kibana et Elasticsearch, si besoin décommentez les lignes suivantes :

```
packetbeat.interfaces.device: any  
  
...  
  
...
```

```
output.elasticsearch:
```

```
  hosts: ["localhost:9200"]
```

```
  username: "" #(si pas de login/mot de passe ne rien mettre)
```

```
  password: "" #(si pas de login/mot de passe ne rien mettre)
```

```
...
```

```
setup.kibana:
```

```
  host: "localhost:5601"
```

```
...
```

```
packetbeat.protocols:
```

```
- type: icmp
```

```
  enabled: false
```

```
- type: dns
```

```
  ports: [53]
```

```
- type: http
```

```
  ports: [80]
```



```
- type: mysql  
  
ports: [3306]Copier
```

Ci-dessous une explication détaillée sur **les options de configuration Packetbeat** utilisées:

- **packetbeat.interfaces.device** : ici on détermine quelle interface réseau surveiller. Dans notre cas, nous allons écouter tous les paquets envoyés ou reçus par le serveur, mais vous pouvez choisir le nom d'une interface réseau spécifique si besoin.
- **packetbeat.protocols** : dans cette section Protocoles, nous devons configurer les ports sur lesquels Packetbeat peut trouver chaque protocole. Habituellement, les valeurs par défaut dans le fichier de configuration suffiront, mais si vous utilisez des ports non standard, c'est l'endroit pour les ajouter. Dans notre cas notre application est desservie par un serveur web et une base de données MySQL.
- **setup.kibana** : pour que les tableaux de bord fonctionnent, nous devons spécifier le point de terminaison Kibana. Vous devrez entrer l'URL de votre hôte Kibana et vos informations d'identification (nom d'utilisateur/mot de passe) si nécessaire.
- **output.elasticsearch** : spécifie la sortie à laquelle nous envoyons les métriques Packetbeat. Nous utilisons Elasticsearch, vous devrez donc fournir l'hôte, le protocole et les informations d'identification Elasticsearch si nécessaire.

Une fois terminé, relancez Packetbeat:

```
sudo systemctl restart packetbeatCopier
```

Visualisation des dashboards

Avant de visualiser nos dashboards, vérifiez au préalable qu'Elasticsearch et Kibana sont en cours d'exécution et qu'Elasticsearch est prêt à recevoir des données de Packetbeat:

```
sudo systemctl status elasticsearch kibanaCopier
```

Résultat :

```
● elasticsearch.service - Elasticsearch  
  
   Loaded: loaded (/lib/systemd/system/elasticsearch.service;  
disabled; vendor preset: enabled)  
  
   Active: active (running) since Wed 2021-07-14 12:29:01 CEST; 1h  
29min ago
```

```
• kibana.service - Kibana
```

```
Loaded: loaded (/etc/systemd/system/kibana.service; disabled;  
vendor preset: enabled)
```

```
Active: active (running) since Wed 2021-07-14 12:28:42 CEST; 1h  
30min ago
```

Ensuite lancez la sous-commande `setup` pour charger les tableaux de bord dans Kibana (la commande peut prendre un peu de temps pour se terminer):

```
sudo packetbeat setup
```

Résultat :

```
Overwriting ILM policy is disabled. Set `setup.ilm.overwrite: true`  
for enabling.
```

```
Index setup finished.
```

```
Loading dashboards (Kibana must be running and reachable)
```

```
Loaded dashboards
```

Afin de charger quelques données pour Packetbeat, visitez plusieurs fois la page <http://localhost> et ajoutez-y quelques articles :

← → ↻ 🏠 🌐 http://localhost

Mon app Packetbeat Accueil Articles

Articles

Nouveau article

Titre *

test

Nom de l'auteur *

test

Contenu *

test test test

Envoyer

Vous pouvez également simuler plusieurs visites, en utiliser la commande `curl` avec une boucle `for` :

```
for i in {1..20}; do curl http://localhost -s > /dev/null; doneCopier
```

Une fois les données chargées, tout pour Metricbeat pour découvrir vos logs, rendez-vous dans <http://localhost:5601/> et sur le menu à gauche cliquez sur Discover:



Home



Home

Recently viewed



No recently viewed items



Kibana



Discover

Dashboard

Canvas

Maps

Machine Learning

Visualize



Observability



Logs

Metrics

APM

Uptime



Security



SIEM

Management



Dev Tools

Stack Monitoring

Stack Management



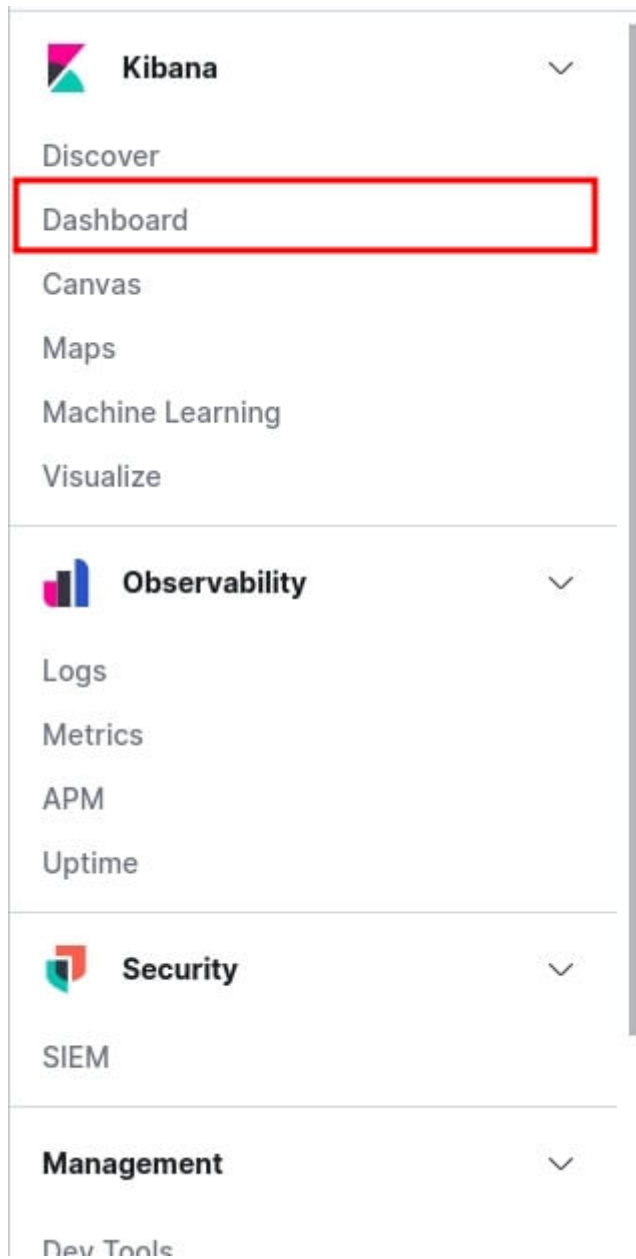
Dock navigation

Choisissez ensuite le pattern index **packetbeat-*** pour visualiser les logs de Packetbeat:

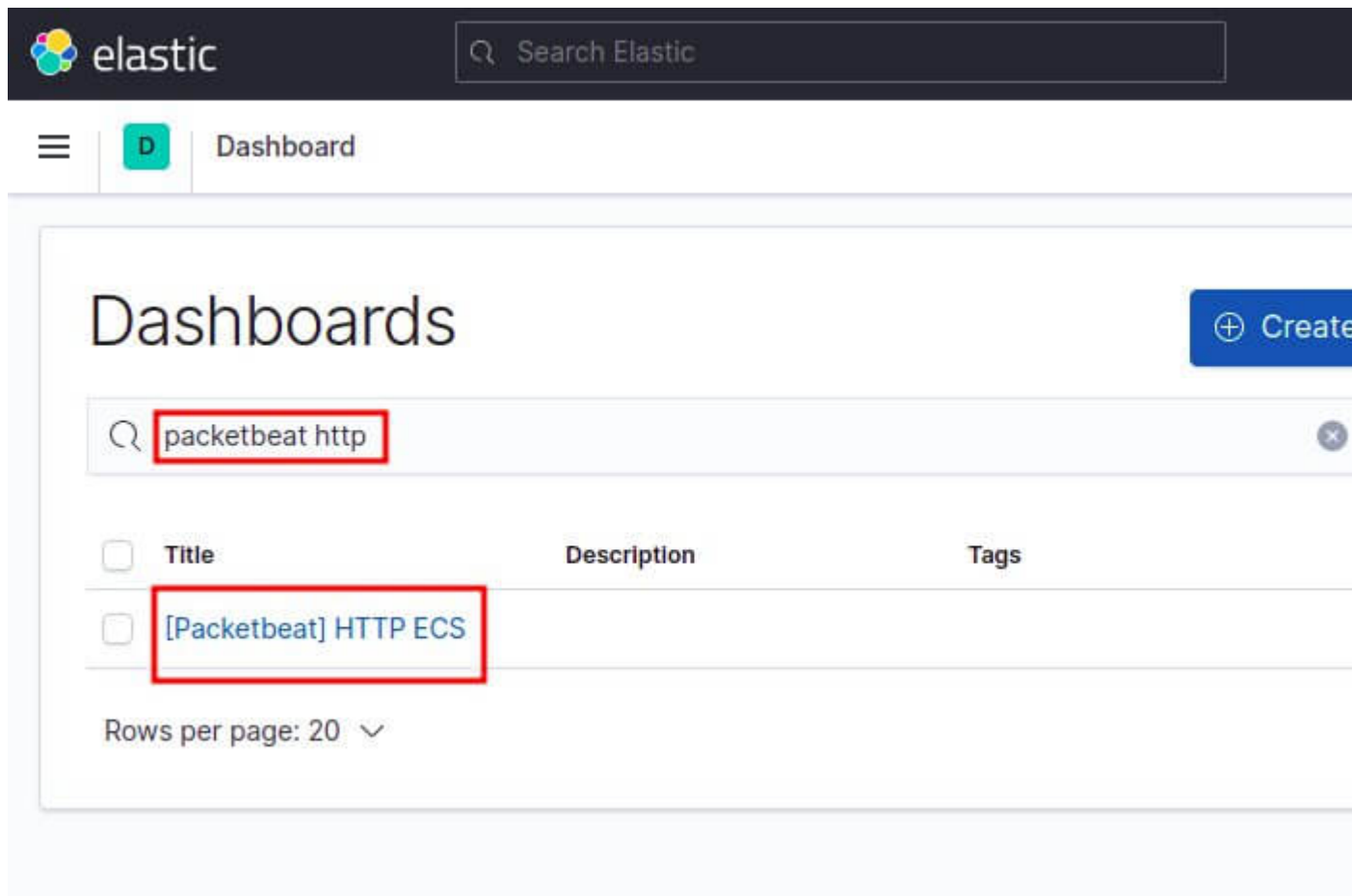
The screenshot shows the Elastic Discover interface. At the top, the Elastic logo and a search bar are visible. Below the search bar, there's a filter section where 'packetbeat-*' is selected and highlighted with a red box. To the right of the filter, it shows '27,051 hits' and a time range of 'Jul 14, 2021 @ 13:46:25.732 - Jul 14, 2021'. Below the filter, there's a list of available fields including '_id', '_index', '_score', '_type', '@timestamp', 'agent.ephemeral_id', 'agent.hostname', 'agent.id', 'agent.name', 'agent.type', 'agent.version', 'bytes_in', 'bytes_out', and 'destination.bytes'. On the right side, there's a bar chart showing the count of hits over time, and a table of documents with columns for 'Time' and 'Document'.

Time	Document
> Jul 14, 2021 @ 14:01:10.000	@timestamp: Jul 14, 2021 @ 14:01:10.000 99d9-4117-8bc0-d59 agent.id: 377d3ef ThinkPad-T14-Gen-1 bytes_in: 260.4KB
> Jul 14, 2021 @ 14:01:10.000	@timestamp: Jul 14, 2021 @ 14:01:10.000 99d9-4117-8bc0-d59 agent.id: 377d3ef ThinkPad-T14-Gen-1 bytes_in: 9.7MB

L'étape suivante est de visualiser notre tableau de bord afin de visualiser la collection de visualisations en temps réel issue par notre expéditeur Packetbeat. Pour ce faire, sur le menu à gauche cliquez sur Dashboard:



Recherchez et cliquez sur les Dashboards de Packetbeat:



Vous obtenez ainsi plusieurs dashboards. Le premier type nommé "HTTP Transactions" ressemble à l'image ci-dessous et vous donne le taux de code de réponse et requête HTTP:

Search

KQL

Last 15 minutes

+ Add filter

Navigation [Packetbeat] ECS

Packetbeat:

[Overview](#)

[Network Flows](#)

[DNS Overview](#) | [Tunneling](#)

[DHCPv4 Transactions](#)

[TLS Overview](#)

[HTTP transactions](#)

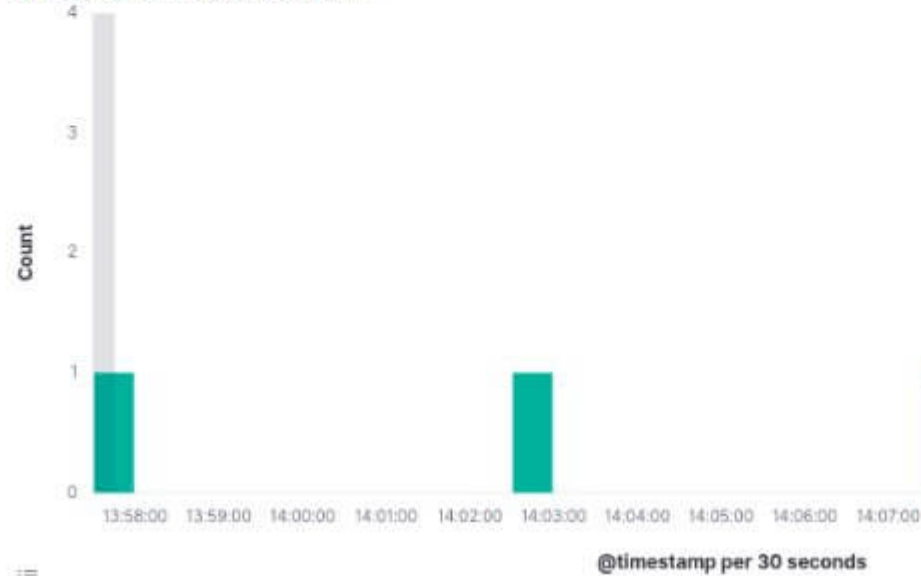
Databases: [MySQL](#) | [PostgreSQL](#) | [MongoDB](#) | [Cassandra](#)

RPC: [Thrift](#)

Storage: [NFS](#)

Total number of HTTP transactions [Pack...

HTTP Transactions [Packetbeat] ECS



HTTP status codes for the top queries [Packetbeat] ECS

7

Count

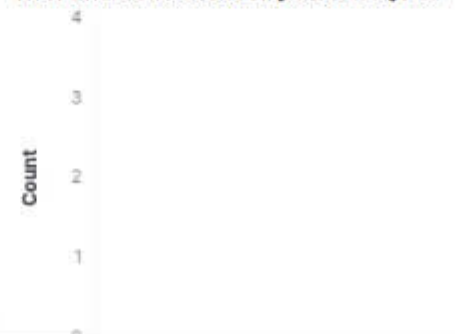
GET /computeMetadata/v1/project/numeric-project-id:...

GET /: HTTP Qu

HTTP error codes [Packetbeat] ECS



HTTP error codes evolution [Packetbeat] ECS



Dans le second nommé "Network Flows", vous avez des informations sur le flux de réseau avec le nombre de connexions dans le temps et les hôtes créant/recevant du trafic incluant également leur consommation réseaux:

Search

KQL

Last 15 m

+ Add filter

Navigation [Packetbeat] ECS

Packetbeat:

[Overview](#)

[Network Flows](#)

[DNS Overview](#) | [Tunneling](#)

[DHCPv4 Transactions](#)

[TLS Overview](#)

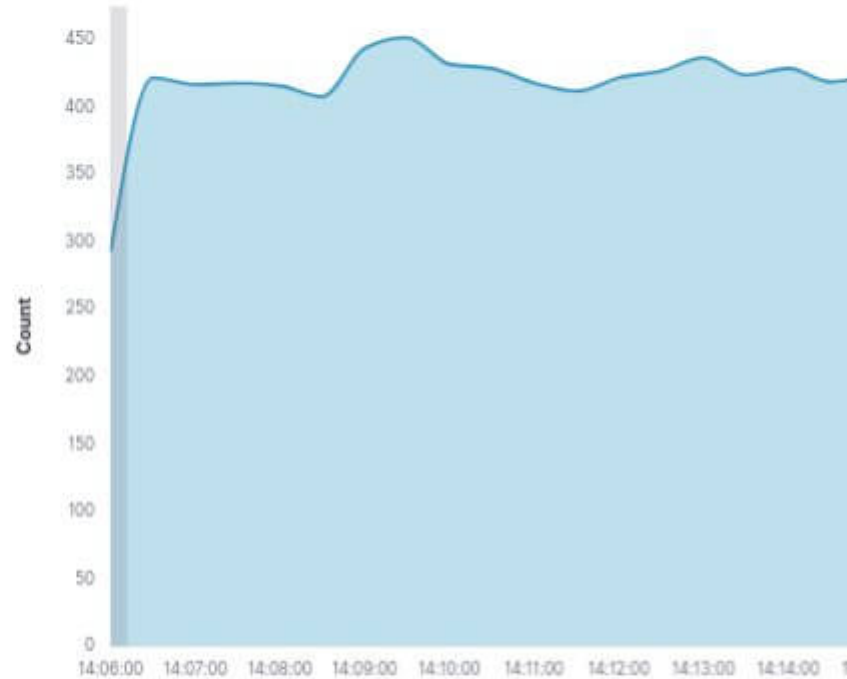
[HTTP transactions](#)

Databases: [MySQL](#) | [PostgreSQL](#) | [MongoDB](#) | [Cassandra](#)

RPC: [Thrift](#)

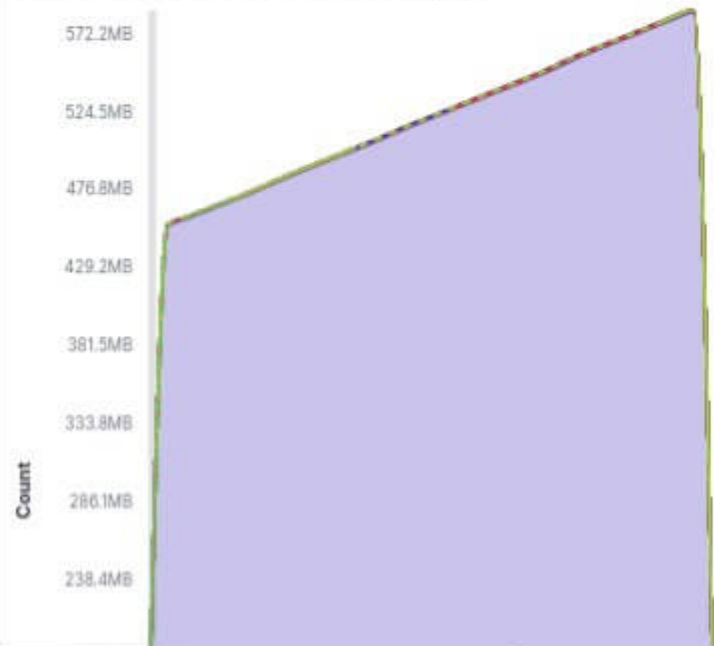
Storage: [NFS](#)

Connections over time [Packetbeat Flows] ECS

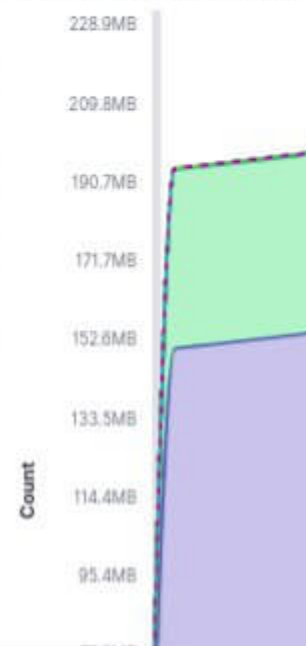


@timestamp per 30 s

Top Hosts Creating Traffic [Packetbeat Flows] ECS



Top Hosts Receiving Traffic [Packetbeat Flows] ECS



Dans le troisième nommé "DNS Overview", vous avez un résumé sur les requêtes DNS avec le temps de réponse moyen, le type de question de votre serveur DNS, histogramme du temps de réponse DNS, etc... :



+ Add Filter

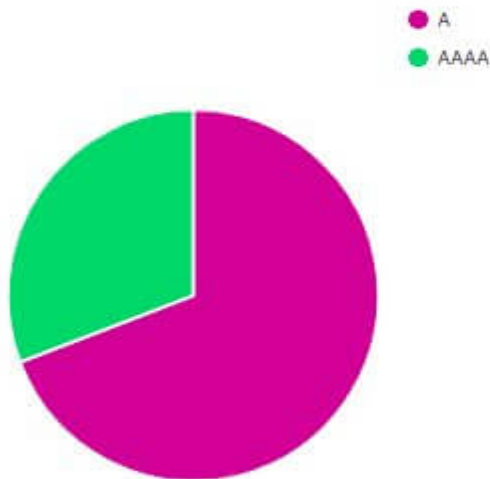
DNS Query Summary [Packetbeat] ECS

266
Count**12.5KB**
Client Bytes**24.7KB**
Server Bytes**16.2**
Avg Response Time (ns)

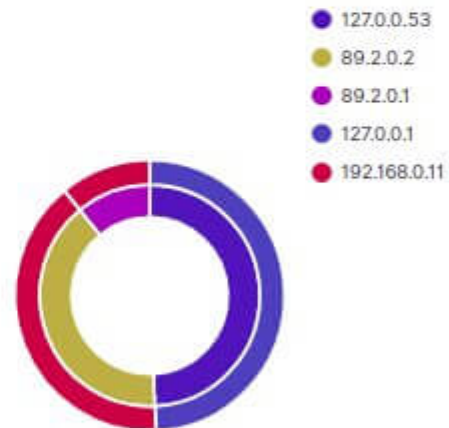
DNS Requ



DNS Question Types [Packetbeat] ECS



DNS Client and Servers Pie Chart [Packetbeat] ECS



DNS Min/M



DNS Top 10 Questions [Packetbeat] ECS

Export

Question	Count
vortex.data.microsoft.com	24
global.vortex.data.trafficmanager.net	15
connectivity-check.ubuntu.com	14

DNS Resp

Exp

Respon

NOERR

NXDOM

Dans le quatrième nommé "Databases: MySQL", vous avez le nombre du type de requête SQL utilisé (SELECT, INSERT, etc ...), le débit de votre serveur MySQL, etc... :



+ Add Filter

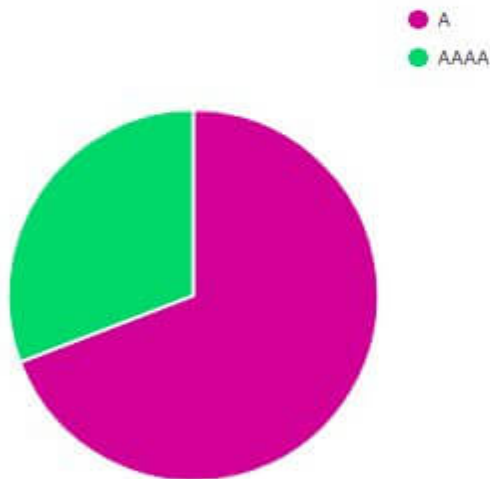
DNS Query Summary [Packetbeat] ECS

266
Count**12.5KB**
Client Bytes**24.7KB**
Server Bytes**16.2**
Avg Response Time (ns)

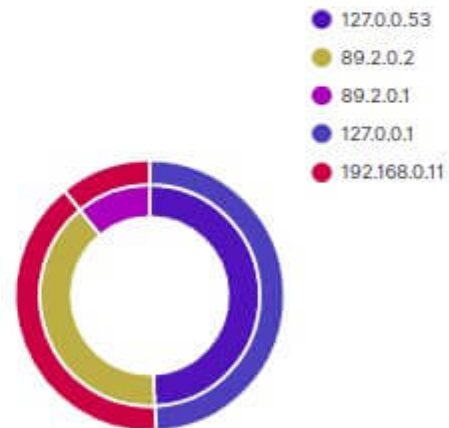
DNS Requ



DNS Question Types [Packetbeat] ECS



DNS Client and Servers Pie Chart [Packetbeat] ECS



DNS Min/M



DNS Top 10 Questions [Packetbeat] ECS

Export

Question	Count
vortex.data.microsoft.com	24
global.vortex.data.trafficmanager.net	15
connectivity-check.ubuntu.com	14

DNS Resp

Exp

Respon

NOERR

NXDOM

Vous avez également d'autres tableaux de bord prêt à l'emploi que vous pouvez visualiser par vous-même.

Conclusion

Maintenant que les métriques réseaux de votre application sont centralisées grâce à Packetbeat, et que vous êtes en mesure de les visualiser avec Kibana, vous devriez pouvoir voir ce que font vos serveurs en un coup d'œil.

Grâce à Packetbeat nous avons pu **avoir rapidement des tableaux de bord de surveillance du réseau opérationnel**, nous donnant une idée en temps réel des paquets transmis sur le fil.

Découverte et utilisation d'Elasticsearch

Dans cet article, nous allons découvrir et apprendre à utiliser Elasticsearch en effectuant des opérations de création, récupération, recherche, modification et suppression de données

Introduction

Elasticsearch (parfois surnommé "ES") est le cœur central de la plate-forme d'analyse de logs la plus populaire à l'heure actuelle à savoir la pile ELK (Elasticsearch, Logstash et Kibana). Dans ce chapitre dédié à Elasticsearch, je vais vous fournir à vous nouveaux utilisateurs **les connaissances et les outils nécessaires pour commencer à utiliser Elasticsearch**.

Qu'est-ce qu'Elasticsearch ?

Mais en fait, **c'est quoi exactement Elasticsearch ?** Un indexeur ? Un moteur de recherche ? Une base de données ? Une solution de Big Data ? La vérité c'est que toutes ces réponses sont correctes et c'est ce qui fait l'intérêt d'Elasticsearch.

En effet, au fil des ans, Elasticsearch a été utilisé pour un nombre croissant de cas d'utilisation, allant de la simple recherche sur un site Web jusqu'à la collecte et l'analyse de données, d'analyses décisionnelles etc ...

Il a été initialement publié en 2010, c'est un outil d'analyse de données moderne basé sur le projet **Apache Lucene**. Entièrement open source et construit avec Java, il fait partie de la famille des **bases de données NoSQL**. Cela signifie qu'il stocke les données de manière non structurées et que vous ne pouvez pas utiliser des requêtes purement SQL pour les interroger.

L'intérêt de ce type de base de données et notamment celui d'Elasticsearch est qu'il vous permet de stocker, rechercher et analyser d'énormes volumes de données rapidement et en temps quasi réel et de vous retourner des réponses en quelques millisecondes.

Ainsi, Elasticsearch est capable d'obtenir des réponses de recherche rapides car au lieu de rechercher le texte directement, il recherche un index. Il utilise une structure basée sur des Documents (nous verrons un peu plus loin ci-dessous la définition d'un Document) au lieu de tables et de schémas et est livré avec des API REST étendues pour stocker et rechercher les données. À la base, vous pouvez considérer Elasticsearch comme un serveur capable de traiter des requêtes JSON et de vous restituer les données sous le format JSON.

Dans le cadre de l'analyse des données, Elasticsearch est utilisé avec les autres composants de la pile ELK (Logstash et Kibana) et joue le rôle d'indexation et de stockage des données.

Information

Ce chapitre sur Elasticsearch pourrait également être considéré comme un **tutoriel NoSQL**. Cependant, contrairement à la plupart des bases de données NoSQL, Elasticsearch met fortement l'accent sur les capacités et les fonctionnalités de recherche, à tel point que le moyen le plus simple d'obtenir des données d'ES est de les rechercher à l'aide de l'API étendue d'Elasticsearch que nous verrons plus loin dans ce chapitre.

Les concepts et terminologies d'Elasticsearch

Pour mieux comprendre le fonctionnement d'Elasticsearch, abordons au préalable quelques **concepts et lexiques de base d'Elasticsearch** et sur la façon dont il organise les données et ses composants backends.

Les Clusters

Nous retrouverons tout d'abord le cluster, qui est un ensemble d'un ou plusieurs nœuds Elasticsearch (serveurs). Il peut comprendre autant de nœuds que vous le souhaitez, ce qui le rend extrêmement évolutif.

La collection de nœuds contient toutes les données du cluster, et le cluster fournit une capacité d'indexation et de recherche sur tous les nœuds. En pratique, cela signifie que lorsque vous effectuez des recherches, vous n'avez pas à vous soucier d'un nœud en particulier sur lequel la donnée sera stockée.

Les Nodes (nœuds)

Un node représente un serveur unique qui stocke des données consultables et fait partie d'un cluster. Les nœuds participent aux capacités d'indexation et de recherche d'un cluster, ce qui signifie que lorsque des opérations sont effectuées sur le cluster, les nœuds collaborent pour répondre aux demandes.

Dans un cluster, différentes responsabilités sont attribuées aux différents types de nœuds:

- **Master nodes** : en charge de la gestion à l'échelle du cluster et des actions de configuration telle que l'ajout et la suppression de nœuds.
- **Data nodes** : stocke les données et exécute les opérations liées aux données telles que la recherche et l'agrégation.
- **Client nodes** : transmet les demandes de cluster au nœud maître et les demandes liées aux données aux Data nodes.
- **Tribe nodes** : agissent comme un Client node, effectuant des opérations de lecture et d'écriture sur tous les nœuds du cluster.
- **Ingestion nodes** : utilisés pour le prétraitement des Documents avant l'indexation (nouveau dans Elasticsearch 5.0).
- **Machine Learning Nodes** : Ce sont des nœuds disponibles sous la licence de base d'Elastic qui permettent des tâches d'apprentissage automatique.

Tous les nœuds sont capables par défaut d'être à la fois des Master nodes, Data nodes, Client nodes, Tribe nodes, Ingestion nodes ou Machine Learning Nodes (c'est le cas pour un cluster

à un seul nœud). Cependant Il est recommandé de distinguer chaque nœud par un seul type, d'autant plus que les clusters grossissent.

Dans un environnement de développement ou de test, vous pouvez configurer plusieurs nœuds sur un seul serveur. En production, cependant en raison du nombre de ressources qu'un nœud Elasticsearch consomme, il est recommandé d'exécuter chaque node Elasticsearch sur un serveur distinct.

Les Fields (champs)

Les fields sont la plus petite unité de données individuelle dans Elasticsearch. Chaque field a un type de données défini et contient une seule donnée. Ces types de données incluent les types de données les plus communs (strings, numbers, dates, booleans), des types de données plus complexes (object and nested), des types de données géographiques (get_point et geo_shape) et les types de données spécialisés (token_count, join, rank_features, dense_vector, etc ..).

Il existe vraiment différentes variétés types de fields et de façons de les gérer, que vous pouvez retrouver sur cette [page](#).

Les Documents

Les Documents sont des objets JSON qui sont stockés dans un index Elasticsearch et sont considérés comme l'unité de base de stockage. Dans le monde des bases de données relationnelles, les Documents peuvent être comparés à une ligne dans une table.

Les données dans les Documents sont définies avec des fields composés de clés et de valeurs. Une clé est le nom du field et une valeur peut être un élément de plusieurs types différents, comme une chaîne de caractères, un nombre, un booléen, un autre objet ou un tableau de valeurs.

Les Documents contiennent également des champs réservés qui constituent les métadonnées du Document tels que :

- **_index**: l'index où réside le Document.
- **_type**: le type que le Document représente.
- **_id**: l'identifiant unique du Document.

Les Documents sont donc l'unité d'information de base qui peut être indexée dans Elasticsearch et ils sont exprimés sous le format JSON. Chaque Document possède donc un identifiant unique et des données composées de fields, qui décrit le type d'entité dont il s'agit. Par exemple, un Document peut représenter un article d'encyclopédie, exemple ci-dessous :

```
{
  "_index" : "votre type d'index",
  "_type" : "votre type d'index",
  "_id" : "1",
  "_source" : {
    "created_at" : "2021-06-07T16:24:32.000Z",
    "title" : "mon article",
    "content" : "ceci est le contenu mon article",
    "author" : "Hatim"
  }
}
```

}Copier

Les Index

Les index, sont la plus grande unité de données dans Elasticsearch qui contiennent un ou plusieurs Documents et peuvent être comparés à une base de données dans le monde des bases de données relationnelles.

Poursuivant notre exemple sur les articles d'encyclopédie, vous pourriez avoir un index nommé "articles" contenant toutes les données liées à vos articles représentés sous forme de Documents qui contiennent des fields de type string sur le titre, le contenu et l'auteur, et un autre index nommé "utilisateurs" avec toutes les données liées à vos utilisateurs avec des Documents qui contiennent des fields de type string sur leur email, nom d'utilisateur et mot de passe.

Vous pouvez avoir autant d'index définis dans Elasticsearch que vous le souhaitez. Ceux-ci contiendront à leur tour des Documents propres à chaque index.

Les types (dépréciés)

Dans un index se trouvent des types de Documents. Un type représente une catégorie de Documents similaires. Un type se compose d'un nom et d'un Mapping, et où le Mapping n'a pas besoin d'être explicitement défini. Vous pouvez penser à un type équivalant à une table dans une base de données relationnelle telle que MySQL. Un index peut avoir un ou plusieurs types, et chacun peut avoir son propre Mapping.

Attention

Différents types de documents étaient autorisés dans un seul index dans la version 6 d'Elasticsearch, mais cette fonctionnalité a été supprimée car cela a conduit à des problèmes divers.

Un seul type de Document est autorisé par index dans la version 7. **La fonctionnalité devrait être complètement supprimée dans la version 8 d'Elasticsearch.** Voici l'explication des développeurs dans leur [blog](#).

Les Mappings

Un type de Document a un Mapping similaire au schéma d'une table dans une base de données relationnelle. Il décrit les champs qu'un Document d'un type donné peut avoir avec leurs types de données, tels qu'une chaîne de caractères, entier, date, etc... Ils ont également des informations sur la façon dont les fields doivent être indexés et comment ils doivent être stockés par Lucene.

Grâce au mappage dynamique, il est facultatif de définir un mappage avant d'ajouter des Documents à un index. Si aucun mappage n'est défini, il sera déduit automatiquement lors de l'ajout d'un Document, en fonction de ses données.

Les Shards (fragments)

Nous allons maintenant discuter d'un nouveau terme appelé shards , c'est également un terme qui existe dans les bases de données relationnelles. Vous avez peut-être entendu parler du

concept de "partitionnement" d'une base de données ? Ne vous inquiétez pas si ce n'est pas le cas.

En effet, la taille de l'index est une cause fréquente de plantage d'Elasticsearch. Comme il n'y a pas de limite au nombre de Documents que vous pouvez stocker sur chaque index, un index peut occuper alors une quantité d'espace disque qui dépasse les limites de stockage d'un nœud. Dès qu'un index approche de cette limite, l'indexation commencera à échouer. Une façon de contrer ce problème consiste à diviser les index horizontalement en morceaux appelés shards (fragments) .

Ceci est utile si un index contient plus de données que le matériel d'un nœud ne peut en stocker. Un exemple pourrait être qu'un index contienne 1 téraoctet de données, mais que le nœud ait un disque dur de seulement 500 gigaoctets. Une shard peut ensuite être créée et stockée sur un autre nœud disposant de suffisamment d'espace pour cela.

Un shard est un index entièrement fonctionnel et indépendant et peut être stocké sur n'importe quel nœud au sein d'un cluster. Le nombre de shards peut être spécifié lors de la création d'un index, mais il est par défaut de 5.

Les shards permettent aussi une mise à l'échelle horizontale par volume de contenu, c'est-à-dire par espace d'index. De plus, le sharding permet de distribuer et de paralléliser les opérations entre les shards, ce qui augmente les performances d'un cluster Elasticsearch.

Les Replicas (répliques)

Alors que les shards améliorent l'évolutivité du volume de contenu pour les index, les Replicas garantissent une haute disponibilité. Une réplique est une copie d'un shard, qui peut prendre le relais en cas de défaillance d'un shard ou d'un nœud.

Un Replica ne réside jamais sur le même nœud que la shard d'origine, ce qui signifie que si le nœud donné échoue, le Replica sera disponible sur un autre nœud. Les répliques permettent également de mettre à l'échelle le volume de recherche, car les répliques peuvent gérer les requêtes de recherche.

Par défaut, Elasticsearch ajoute cinq shards principaux et une réplique pour chaque index, ce qui signifie que, sauf configuration contraire, qu'il y aura une réplique pour chaque shard principale, soit cinq au total.

D'autres concepts

Ce sont les principaux concepts que vous devez comprendre lorsque vous commencez à utiliser Elasticsearch, mais il existe également d'autres composants et termes sur Elasticsearch. Je ne peux pas tous les couvrir, je vous recommande donc de vous référer à la [page des terminologies Elasticsearch](#) pour plus d'informations.

Utilisation d'Elasticsearch

Installation et configuration

Vous trouverez mon tutoriel sur l'installation d'Elasticsearch dans mon [article](#).

Les configurations Elasticsearch se font à l'aide d'un fichier de configuration dont l'emplacement dépend de votre système d'exploitation. Dans ce fichier, vous pouvez configurer les paramètres généraux (par exemple le nom du nœud), les paramètres réseau (par exemple l'hôte et le port), l'emplacement des données à stocker, la mémoire, les fichiers de logs, etc.

À des fins de développement et de test, les paramètres par défaut suffiront, mais il est recommandé de faire des recherches sur les paramètres que vous devez définir manuellement avant de passer en production. Par défaut le fichier de configuration d'Elasticsearch est le suivant:

Autre chose, Elasticsearch ne s'exécutera pas automatiquement après l'installation et vous devrez le démarrer manuellement. La façon dont vous exécutez Elasticsearch dépend de votre système spécifique. Sur la plupart des systèmes Linux et Unix, vous avez juste à démarrer le service Elasticsearch (l'initialisation peut prendre un peu de temps) :

```
sudo systemctl start elasticsearch
```

Pour confirmer que tout fonctionne correctement, pointez simplement avec la commande `curl` ou votre navigateur sur l'url et le port suivant <http://localhost:9200>, et vous devriez voir quelque chose comme la sortie suivante :

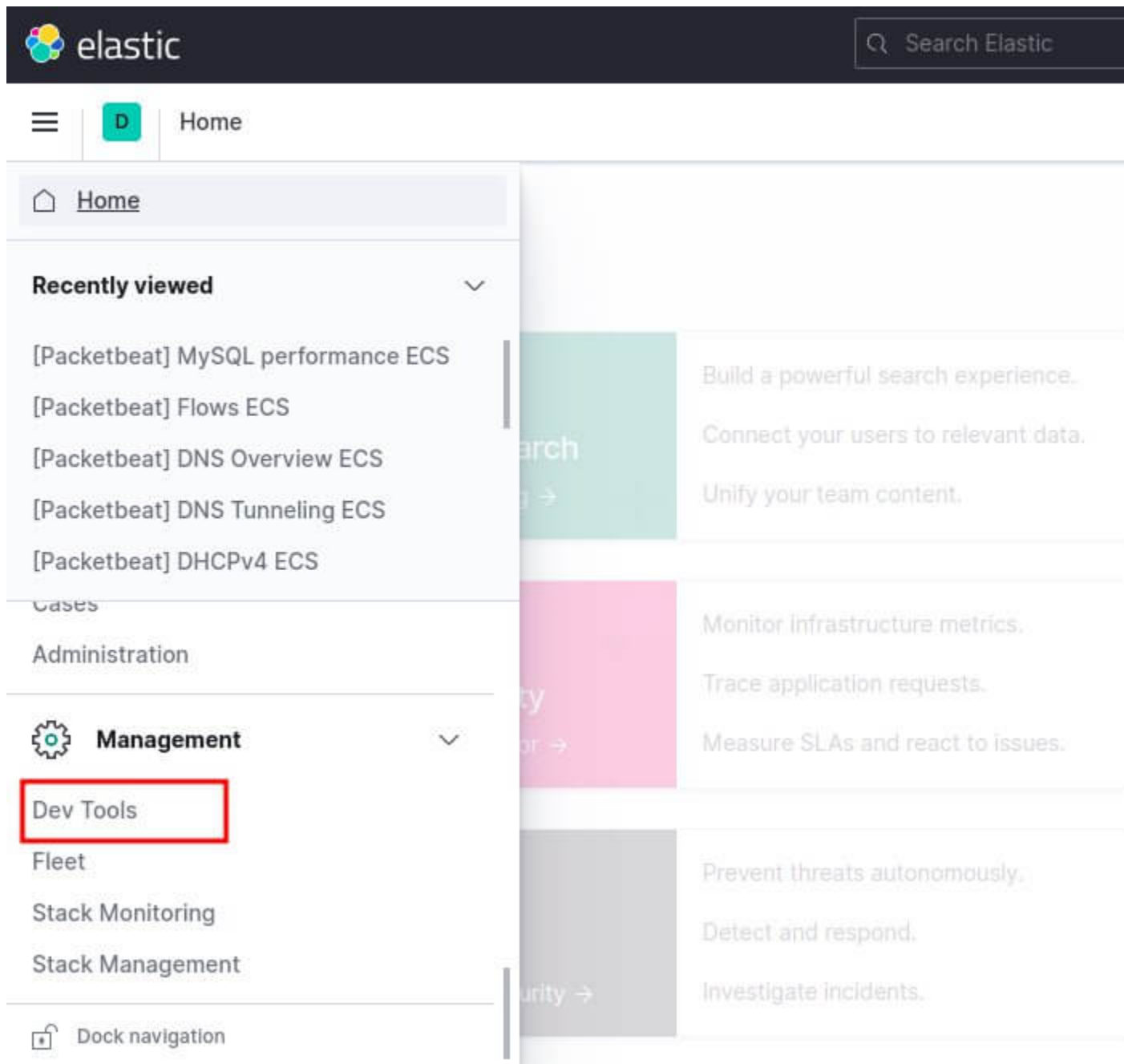
```
curl http://localhost:9200
```

Résultat :

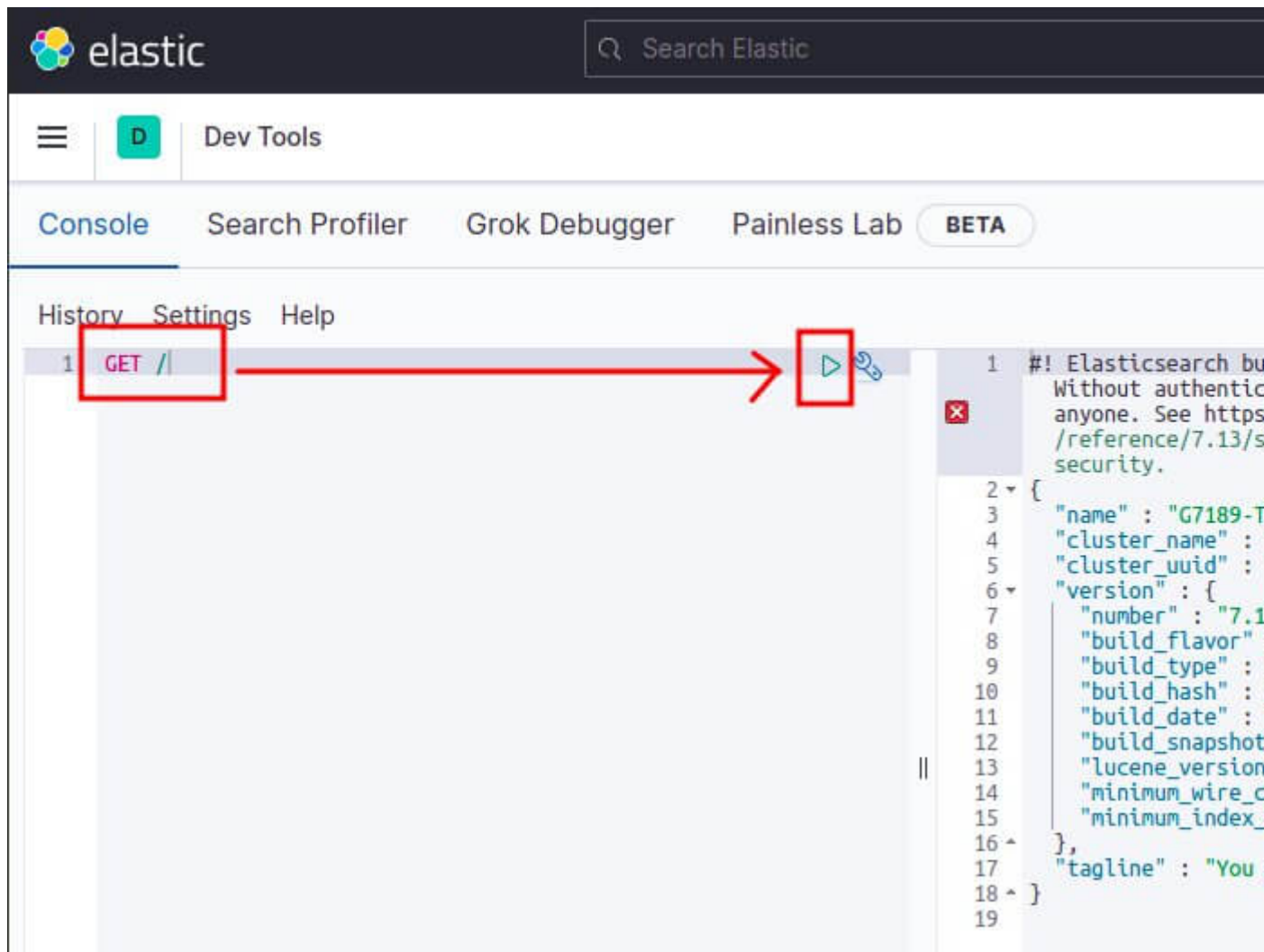
```
{
  "name" : "G7189-ThinkPad-T14-Gen-1",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "0Aik7GWhRL006xKaseK7iw",
  "version" : {
    "number" : "7.13.3",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "5d21bea28db1e89ecc1f66311ebdec9dc3aa7d64",
    "build_date" : "2021-07-02T12:06:10.804015202Z",
    "build_snapshot" : false,
    "lucene_version" : "8.8.2",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Copier

Vous pouvez également **communiquer avec Elasticsearch depuis la console Kibana**. Pour cela accéder à la page Kibana et depuis le menu gauche cliquez sur "Dev Tools":



Et exécutez une requête de type GET sur l'API Elasticsearch afin de récupérer des informations sur le node :



Si jamais vous rencontrez des problèmes d'initialisation, vérifiez vos logs pour **déboguer le processus d'exécution d'Elasticsearch** :

```
sudo journalctl -f -u elasticsearchCopier
```

Pour initialiser le service Elasticsearch à chaque démarrage de la machine, lancez la commande suivante :

```
sudo systemctl enable elasticsearchCopier
```

Création de données dans Elasticsearch

L'indexation est le processus d'ajout de données dans Elasticsearch. En effet, lorsque vous fournissez des données à Elasticsearch, les données sont placées dans les index Apache Lucene . Cela est logique, car Elasticsearch utilise les index Lucene pour stocker et récupérer ses données. Bien que vous n'ayez pas besoin d'en savoir beaucoup sur Lucene, il est utile de

savoir comment cela fonctionne lorsque vous commencez à devenir plus expert avec Elasticsearch.

Elasticsearch se comporte comme une API REST, vous pouvez donc utiliser méthode de type POST ou PUT pour y ajouter des données. Vous utiliserez la méthode PUT, lorsque vous connaissez déjà l'id de l'élément de données que vous souhaitez spécifier, ou la méthode POST si vous souhaitez qu'Elasticsearch génère pour vous un id automatiquement pour l'élément de données.

Dans notre exemple, nous allons établir un système d'articles d'encyclopédie. Nous allons donc créer un index qu'on nommera "articles" qui contiendra un Document avec des fields de type string sur le nom de l'auteur, le titre, la catégorie et le contenu de l'article ainsi que la date de publication qui sera de type date. Nous obtiendrons ainsi la requête suivante:

```
curl -X POST 'http://localhost:9200/articles/_doc?pretty' -H  
'Content-Type: application/json' -d '
```

```
{  
  
  "created_at": "2021-06-07T16:24:32.000Z",  
  
  "title": "mon article sur Elasticsearch",  
  
  "content": "ceci est le contenu de mon article sur  
Elasticsearch",  
  
  "category": "elasticsearch",  
  
  "author": "Hatim"  
  
}
```

'Copier

Résultat :

```
{  
  
  "_index" : "articles",  
  
  "_type" : "_doc",
```

```
"_id" : "0YZ7uXoBzgM1NBNTNXt1",

"_version" : 1,

"result" : "created",

"_shards" : {

  "total" : 2,

  "successful" : 1,

  "failed" : 0

},

"_seq_no" : 3,

"_primary_term" : 1

}Copier
```

Comme vu sur le résultat, l'id est généré automatiquement pour nous. Pour le faire depuis la méthode PUT il suffit juste d'ajouter l'id à votre entrée:

```
curl -X PUT 'http://localhost:9200/articles/_doc/1?pretty' -H
'Content-Type: application/json' -d '

{

  "created_at": "2021-05-07T16:24:32.000Z",

  "title": "mon second article sur Elasticsearch",

  "content": "ceci est le contenu de mon second article sur
Elasticsearch",

  "category": "elasticsearch",

  "author": "Hatim"

}
```

'Copier

Résultat :

```
{
  "_index" : "articles",
  "_type" : "_doc",
  "_id" : "1",
  "_version" : 1,
  "result" : "created",
  "_shards" : {
    "total" : 2,
    "successful" : 1,
    "failed" : 0
  },
  "_seq_no" : 3,
  "_primary_term" : 1
}
```

}Copier

Information

Le paramètre **pretty** de votre requête renvoie un JSON assez formaté (à utiliser uniquement pour le débogage !). Une autre option consiste à définir **?format=yaml** ce qui entraînera le retour du résultat au format YAML (parfois) plus lisible.

Pour ajouter plusieurs Documents en une seule requête, utilisez alors le paramètre **_bulk** en point de terminaison de votre requête. les fields de vos Documents au format JSON doivent

toujours être sur la même ligne et délimités par une nouvelle ligne. Chaque ligne doit se terminer par un caractère de nouvelle ligne, y compris la dernière ligne.

```
curl -X POST "localhost:9200/articles/_bulk?pretty" -H 'Content-Type: application/json' -d'

{ "create": { } }

{ "created_at": "2021-08-07T16:24:32.000Z", "title": "mon article sur la stack ELK", "content": "ceci est le contenu de mon article sur stack ELK", "category": "ELK", "author": "Hatim" }

{ "create": { } }

{ "created_at": "2021-03-07T16:24:32.000Z", "title": "mon second article sur la stack ELK", "content": "ceci est le contenu de mon second article sur stack ELK", "category": "ELK", "author": "Hatim" }

'Copier
```

Résultat :

```
{

  "took" : 10,

  "errors" : false,

  "items" : [

    {

      "create" : {

        "_index" : "articles",

        "_type" : "_doc",

        "_id" : "zoZ5uXoBzgM1NBNTantg",

        "_version" : 1,
```

```
    "result" : "created",

    "_shards" : {

        "total" : 2,

        "successful" : 1,

        "failed" : 0

    },

    "_seq_no" : 1,

    "_primary_term" : 1,

    "status" : 201

}

},

{

    "create" : {

        "_index" : "articles",

        "_type" : "_doc",

        "_id" : "z4Z5uXoBzgM1NBNTantg",

        "_version" : 1,

        "result" : "created",

        "_shards" : {

            "total" : 2,

            "successful" : 1,
```

```

      "failed" : 0

    },

    "_seq_no" : 2,

    "_primary_term" : 1,

    "status" : 201

  }

}

]
}Copier

```

Récupération de données dans Elasticsearch

Pour voir la liste complète de vos index Elasticsearch, utilisez la requête suivante :

```
curl -X GET 'localhost:9200/_cat/indices?pretty'Copier
```

Résultat :

```

green open .kibana_7.13.3_001
_QiHUUwqQEW3EB5YgEa1zw 1 0      4571      17      3.3mb      3.3mb

yellow open apache-2021.07.12
rk1U3br_TRyUenUu6U3plw 1 1      85        0      124.4kb     124.4kb

green open .apm-agent-configuration
JMLcxwD8QqKj_7ZgQcKSdg 1 0        0        0        208b       208b

yellow open packetbeat-7.13.3-2021.07.14-000001 AlCvCInWRM6-
B3mXUx4mVw 1 1 2770319      0      922mb      922mb

green open .kibana_task_manager_7.13.3_001
ynWJoHQ4TIWrm5W_iy8RIQ 1 0        10 15902      2.3mb      2.3mb

```

```

green open .tasks nmDN_yhTSti1Q-
s2kdQ2aQ 1 0 12 0 69.4kb 69.4kb

yellow open metricbeat-7.13.3-2021.07.13-000001 jHgARIOyRd-
SfFfrWzMPw 1 1 63684 0 29.7mb 29.7mb

green open .kibana-event-log-7.13.3-000001
oS0Qf0ChTTiSiGzPE1aq5Q 1 0 8 0 43.3kb 43.3kb

green open .apm-custom-link
wypmzo85R8uQ2UTcbnyxWQ 1 0 0 0 208b 208b

green open .async-search
6wnbxfSSREm4S1BkTIg09Q 1 0 174 100 1.4mb 1.4mb

yellow open filebeat-7.13.3-2021.07.12-000001
SSm52h8kTeeXcy4pOUM3Jg 1 1 936 0 269.5kb 269.5kb

yellow open articles
VX_kMJDSTagmTVqJ0rmw 1 1 1 0 6.8kb 6.8kb

```

Dans cette sortie, nous retrouvons la liste de l'index que nous avons créé précédemment, un index Kibana et les index créés dans nos précédents chapitres.

Une fois que vous avez indexé vos données dans Elasticsearch, vous pouvez commencer à les récupérer et à les analyser. Encore une fois, via l'API REST Elasticsearch, nous utilisons la méthode GET.

Pour afficher tous les Documents d'un index dans Elasticsearch nous utiliserons la requête GET avec comme point de terminaison le mot `_search` :

```
curl -X GET 'localhost:9200/articles/_doc/_search?pretty'
```

Résultat :

```

{
  "took" : 0,
  "timed_out" : false,
  "_shards" : {

```



```
"total" : 1,

"successful" : 1,

"skipped" : 0,

"failed" : 0

},

"hits" : {

  "total" : {

    "value" : 4,

    "relation" : "eq"

  },

  "max_score" : 1.0,

  "hits" : [

    {

      "_index" : "articles",

      "_type" : "_doc",

      "_id" : "1",

      "_score" : 1.0,

      "_source" : {

        "created_at" : "2021-05-07T16:24:32.000Z",

        "title" : "mon second article sur Elasticsearch",

        "content" : "ceci est le contenu de mon second article sur Elasticsearch",
```

```
      "category" : "elasticsearch",

      "author" : "Hatim"

    }

  },

  {

    "_index" : "articles",

    "_type" : "_doc",

    "_id" : "zoZ5uXoBzgM1NBNTantg",

    "_score" : 1.0,

    "_source" : {

      "created_at" : "2021-08-07T16:24:32.000Z",

      "title" : "mon article sur la stack ELK",

      "content" : "ceci est le contenu de mon article sur stack
ELK",

      "category" : "ELK",

      "author" : "Hatim"

    }

  },

  {

    "_index" : "articles",

    "_type" : "_doc",

    "_id" : "z4Z5uXoBzgM1NBNTantg",
```

```
    "_score" : 1.0,

    "_source" : {

        "created_at" : "2021-03-07T16:24:32.000Z",

        "title" : "mon second article sur la stack ELK",

        "content" : "ceci est le contenu de mon second article sur
stack ELK",

        "category" : "ELK",

        "author" : "Hatim "

    }

},

{

    "_index" : "articles",

    "_type" : "_doc",

    "_id" : "0YZ7uXoBzgM1NBNTNXt1",

    "_score" : 1.0,

    "_source" : {

        "created_at" : "2021-06-07T16:24:32.000Z",

        "title" : "mon article sur Elasticsearch",

        "content" : "ceci est le contenu de mon article sur
Elasticsearch",

        "category" : "elasticsearch",

        "author" : "Hatim"

    }

}
```

```
}  
  
]  
  
}  
}Copier
```

Le résultat contient un certain nombre de fields supplémentaires qui décrivent à la fois la recherche et le résultat. Voici un aperçu rapide de ces fields :

- **_took** : le temps en millisecondes que la recherche a pris.
- **_timed_out** : s'il y a eu un Time Out dans votre recherche.
- **_shards** : le nombre de shards Lucene recherchés, et leurs taux de réussite et d'échec.
- **_hits** : les résultats réels, ainsi que les métadonnées pour les résultats.
- **_score** : le score de pertinence de chaque Document trouvé dans votre recherche (plus le score est élevé plus le résultat sera pertinent).

Voici la commande pour récupérer un Document bien particulier via son id :

```
curl -X GET 'localhost:9200/articles/_doc/1?pretty'Copier
```

Résultat :

```
{  
  
  "_index" : "articles",  
  
  "_type" : "_doc",  
  
  "_id" : "1",  
  
  "_version" : 1,  
  
  "_seq_no" : 0,  
  
  "_primary_term" : 1,  
  
  "found" : true,  
  
  "_source" : {  
  
    "created_at" : "2021-05-07T16:24:32.000Z",
```

```
"title" : "mon second article sur Elasticsearch",

"content" : "ceci est le contenu de mon second article sur
Elasticsearch",

"category" : "elasticsearch",

"author" : "Hatim"

}

}Copier
```

Vous avez également la possibilité d'afficher uniquement certains fields de votre Document. Supposons que nous souhaitons récupérer uniquement le nom de l'auteur et le titre de l'article, il faut alors exécuter une requête GET avec le paramètre `_source` suivie des noms des fields à filtrer :

```
curl -X GET
'localhost:9200/articles/_doc/1?pretty&_source=author,title'Copier
```

Résultat :

```
{

  "_index" : "articles",

  "_type" : "_doc",

  "_id" : "1",

  "_version" : 1,

  "_seq_no" : 0,

  "_primary_term" : 1,

  "found" : true,

  "_source" : {

    "author" : "Hatim",
```

```
    "title" : "mon second article sur Elasticsearch"

  }

}Copier
```

Si les métadonnées d'une entrée ne vous intéressent pas, utilisez la commande suivante :

```
curl -X GET
'localhost:9200/articles/_doc/1/_source?pretty&_source=author,title'
Copier
```

Résultat :

```
{

  "author" : "Hatim",

  "title" : "mon second article sur Elasticsearch"

}Copier
```

Rechercher des données dans Elasticsearch

Nous utiliserons également GET pour effectuer des recherches en appelant cette fois-ci le paramètre `_search` comme point de terminaison. Pour le moment récupérant tous nos Documents avec le filtre `match_all`, comme suit :

```
curl -X GET "localhost:9200/articles/_doc/_search?pretty" -H
'Content-Type: application/json' -d'

{

  "query": {

    "match_all": { }

  }

}

'Copier
```

Résultat :

```
{
  "took" : 0,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "articles",
        "_type" : "_doc",
        "_id" : "y4ZSuXoBzgM1NBNTFHtr",
```

```
    "_score" : 1.0,

    "_source" : {

        "created_at" : "2021-07-23 16:21:46",

        "title" : "mon article sur la stack ELK",

        "content" : "ceci est le contenu de mon article sur stack
ELK",

        "category" : "ELK",

        "author" : "Hatim"

    }

},

{

    "_index" : "articles",

    "_type" : "_doc",

    "_id" : "zIZSuXoBzgM1NBNTFHtr",

    "_score" : 1.0,

    "_source" : {

        "created_at" : "2021-07-23 16:21:46",

        "title" : "mon second article sur la stack ELK",

        "content" : "ceci est le contenu de mon second article sur
stack ELK",

        "category" : "ELK",

        "author" : "Hatim "

    }

}
```



```
    },  
  
    etc ...  
  
  ]  
  
}
```

}Copier

Si les métadonnées ne vous intéressent pas, alors vous pouvez utiliser paramètre `filter_path` pour réduire la réponse renvoyée par Elasticsearch. Ce paramètre prend une liste de filtres séparés par des virgules. Exemple :

```
curl -X GET  
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._  
source" -H 'Content-Type: application/json' -d'  
  
{  
  
  "query": {  
  
    "match_all": { }  
  
  }  
  
}
```

'Copier

Résultat :

```
{  
  
  "hits" : {  
  
    "hits" : [  
  
      {  
  
        "_source" : {
```

```
        "created_at" : "2021-05-07T16:24:32.000Z",

        "title" : "mon second article sur Elasticsearch",

        "content" : "ceci est le contenu de mon second article sur
Elasticsearch",

        "category" : "elasticsearch",

        "author" : "Hatim"

    }

},

{

    "_source" : {

        "created_at" : "2021-08-07T16:24:32.000Z",

        "title" : "mon article sur la stack ELK",

        "content" : "ceci est le contenu de mon article sur stack
ELK",

        "category" : "ELK",

        "author" : "Hatim"

    }

},

etc ...

]

}
```

}Copier

Par défaut, la section **hits** de la réponse inclut jusqu'aux 10 premiers Documents qui correspondent à la recherche. Pour augmenter le nombre de Documents de votre recherche utilisez le paramètre **size** suivi du nombre de résultats que vous souhaitez afficher :

```
curl -X GET
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._
source&size=15" -H 'Content-Type: application/json' -d'

{

  "query": {

    "match_all": { }

  }

}

'Copier
```

Vous avez également la possibilité de trier vos résultats grâce au filtre **sort**. Dans notre cas nous souhaitons afficher les articles les plus récents en triant nos résultats par ordre décroissant depuis notre field **created_at** :

```
curl -X GET
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._
source" -H 'Content-Type: application/json' -d'

{

  "query": {

    "match_all": { }

  },

  "sort": [ {"created_at": "desc"} ]

}

'Copier
```

Résultat :

```
{
  "hits" : {
    "hits" : [
      {
        "_source" : {
          "created_at" : "2021-08-07T16:24:32.000Z",
          "title" : "mon article sur la stack ELK",
          "content" : "ceci est le contenu de mon article sur stack
ELK",
          "category" : "ELK",
          "author" : "Hatim"
        }
      },
      {
        "_source" : {
          "created_at" : "2021-06-07T16:24:32.000Z",
          "title" : "mon article sur Elasticsearch",
          "content" : "ceci est le contenu de mon article sur
Elasticsearch",
          "category" : "elasticsearch",
          "author" : "Hatim"
        }
      }
    ]
  }
}
```

```
    },  
  
    {  
  
      "_source" : {  
  
        "created_at" : "2021-05-07T16:24:32.000Z",  
  
        "title" : "mon second article sur Elasticsearch",  
  
        "content" : "ceci est le contenu de mon second article sur  
Elasticsearch",  
  
        "category" : "elasticsearch",  
  
        "author" : "Hatim"  
  
      }  
  
    },  
  
    {  
  
      "_source" : {  
  
        "created_at" : "2021-03-07T16:24:32.000Z",  
  
        "title" : "mon second article sur la stack ELK",  
  
        "content" : "ceci est le contenu de mon second article sur  
stack ELK",  
  
        "category" : "ELK",  
  
        "author" : "Hatim "  
  
      }  
  
    }  
  
  ]  
  
}
```

}Copier

Pour récupérer les Documents contenant une valeur spécifique dans un field vous utilisez le filtre **match** suivi du field et de la valeur à rechercher:

```
url -X GET
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._
source" -H 'Content-Type: application/json' -d'

{

  "query": {

    "match": { "category": "ELK" }

  }

}

'Copier
```

Résultat :

```
{

  "hits" : {

    "hits" : [

      {

        "_source" : {

          "created_at" : "2021-08-07T16:24:32.000Z",

          "title" : "mon article sur la stack ELK",

          "content" : "ceci est le contenu de mon article sur stack
ELK",

          "category" : "ELK",
```

```

        "author" : "Hatim"
    }
},
{
    "_source" : {
        "created_at" : "2021-03-07T16:24:32.000Z",
        "title" : "mon second article sur la stack ELK",
        "content" : "ceci est le contenu de mon second article sur
stack ELK",
        "category" : "ELK",
        "author" : "Hatim "
    }
}
]
}
}Copier

```

Pour faire une recherche sur des fields spécifiques, vous utiliserez le filtre **fields**, comme suit:

```

curl -X GET
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._
source" -H 'Content-Type: application/json' -d'
{
    "query": {
        "multi_match" : {

```

```
    "query": "ELK",

    "fields": [ "title", "content" ]

  }

}

}
```

'Copier

Résultat :

```
{

  "hits" : {

    "hits" : [

      {

        "_source" : {

          "created_at" : "2021-08-07T16:24:32.000Z",

          "title" : "mon article sur la stack ELK",

          "content" : "ceci est le contenu de mon article sur stack ELK",

          "category" : "ELK",

          "author" : "Hatim"

        }

      },

      {

        "_source" : {
```



```
      "created_at" : "2021-03-07T16:24:32.000Z",

      "title" : "mon second article sur la stack ELK",

      "content" : "ceci est le contenu de mon second article sur
stack ELK",

      "category" : "ELK",

      "author" : "Hatim "

    }

  }

]

}
```

}Copier

Nous avons également un autre type de requêtes qu'on surnomme "Range query". Ils permettent d'effectuer une recherche sur une plage de données. Dans notre cas nous souhaitons récupérer les articles créés à partir du 07/07/2021. Pour ce faire, nous utiliserons le filtre **range** avec l'option **gte**:

```
curl -X GET
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._
source" -H 'Content-Type: application/json' -d'

{

  "query": {

    "range": {

      "created_at": {

        "gte": "2021-07-07"

      }

    }

  }

}
```

```
}

},

"fields": [

  "created_at"

],

"sort": [

  {

    "created_at": "desc"

  }

]

}

'Copier
```

Résultat :

```
{

  "hits" : {

    "hits" : [

      {

        "_source" : {

          "created_at" : "2021-08-07T16:24:32.000Z",

          "title" : "mon article sur la stack ELK",

          "content" : "ceci est le contenu de mon article sur stack ELK",
```

```
      "category" : "ELK",  
      "author" : "Hatim"  
    }  
  }  
]  
}
```

}Copier

les autres options disponibles pour ce type de filtres sont les suivantes:

- **gt** : supérieur à.
- **gte** : supérieur ou égal à.
- **lt** : moins de.
- **lte** : inférieur ou égal à.

Nous avons également un autre type de requêtes qu'on surnomme "Bool query". Il permettent de combiner plusieurs recherches sous forme de clause/conditions (correspond au **if** dans un langage de programmation standard) dans une seule et même requête.

Discutons d'abord de la structure générale de la requête Bool :

```
POST _search  
  
{  
  "query": {  
    "bool": {  
      "must": [...],  
      "filter": [...],  
      "must_not": [...],  
      "should": [...]    }  
  }  
}
```

```
}  
  
}  
  
}
```

Voici une explication des différentes options du filtre **bool** :

- **must** : la condition doit apparaître dans les Documents correspondants et contribuera à modifier le score de chaque résultat (correspond au **OU** logique dans un langage de programmation).
- **filtre** : la condition doit apparaître dans les Documents correspondants. Cependant, contrairement à **must**, le score de la requête sera ignoré.
- **should** : la condition doit apparaître dans le document correspondant (correspond au **AND** logique dans un langage de programmation).
- **must_not** : la condition ne doit pas apparaître dans les documents correspondants.

Dans cet exemple nous souhaitons récupérer tous les articles créés par l'auteur "Hatim" et qui ne sont pas de catégorie "elasticsearch" et créés à partir du 03/07/2021. Nous aurons ainsi la requête suivante :

```
curl -X GET  
"localhost:9200/articles/_doc/_search?pretty&filter_path=hits.hits._  
source" -H 'Content-Type: application/json' -d'  
  
{  
  
  "query": {  
  
    "bool": {  
  
      "must": [  
  
        {  
  
          "match": {  
  
            "author": "Hatim"  
  
          }  
  
        },  
  
      ],  
  
    }  
  
  }  
  
}
```

```

    {
      "range": {
        "created_at": {
          "gte" : "2021-03-07"
        }
      }
    },
    "must_not": [
      {
        "match": {
          "category": "elasticsearch"
        }
      }
    ]
  }
}

```

'Copier

Résultat :

```
{
```

```
"hits" : {  
  "hits" : [  
    {  
      "_source" : {  
        "created_at" : "2021-08-07T16:24:32.000Z",  
        "title" : "mon article sur la stack ELK",  
        "content" : "ceci est le contenu de mon article sur stack  
ELK",  
        "category" : "ELK",  
        "author" : "Hatim"  
      }  
    },  
    {  
      "_source" : {  
        "created_at" : "2021-03-07T16:24:32.000Z",  
        "title" : "mon second article sur la stack ELK",  
        "content" : "ceci est le contenu de mon second article sur  
stack ELK",  
        "category" : "ELK",  
        "author" : "Hatim "  
      }  
    }  
  ]  
}
```

```
}
```

```
}Copier
```

Ce type de requêtes qui permettent des recherches plus avancées est nommé **requête DSL**. Nous avons pu voir quelques-unes d'entre elles, mais il existe un large éventail d'options disponibles dans ce type de recherche que vous pouvez combiner et assortir avec différentes options pour obtenir les résultats dont vous avez besoin. Plus d'informations sur la [page officielle ELK](#).

Modifier des données dans Elasticsearch

Pour ajouter un field à un Document déjà existant, utilisez la méthode POST avec le paramètre `_update`. Dans notre exemple, nous allons ajouter un field nommé "is_private" afin de mettre un de nos articles créé précédemment en privé:

```
curl -X POST "localhost:9200/articles/_update/1?pretty" -H 'Content-Type: application/json' -d'
```

```
{  
  
  "doc": {  
  
    "is_private": true  
  
  }  
  
}
```

```
' Copier
```

Résultat :

```
{  
  
  "_index" : "articles",  
  
  "_type" : "_doc",  
  
  "_id" : "1",
```

```
"_version" : 6,

"result" : "updated",

"_shards" : {

  "total" : 2,

  "successful" : 1,

  "failed" : 0

},

"_seq_no" : 8,

"_primary_term" : 1

}Copier
```

Vérifions si notre field s'est bien enregistré dans notre Document:

```
curl -X GET 'localhost:9200/articles/_doc/1/_source?pretty'Copier
```

Résultat :

```
{

  "created_at" : "2021-05-07T16:24:32.000Z",

  "title" : "mon second article sur Elasticsearch",

  "content" : "ceci est le contenu de mon second article sur Elasticsearch",

  "category" : "elasticsearch",

  "author" : "Hatim",

  "is_private" : true

}Copier
```


C'est bien le cas ! D'ailleurs vous pouvez utiliser la même requête pour modifier la valeur d'un field. Dans cet exemple, nous allons passer notre article en mode public:

```
curl -X POST "localhost:9200/articles/_update/1?pretty" -H 'Content-Type: application/json' -d'
```

```
{  
  
  "doc": {  
  
    "is_private": false  
  
  }  
  
}
```

' Copier

Résultat :

```
{  
  
  "_index" : "articles",  
  
  "_type" : "_doc",  
  
  "_id" : "1",  
  
  "_version" : 7,  
  
  "result" : "noop",  
  
  "_shards" : {  
  
    "total" : 0,  
  
    "successful" : 0,  
  
    "failed" : 0  
  
  },  
  
}
```

```
"_seq_no" : 9,  
"_primary_term" : 1  
}Copier
```

Vérifions si notre field s'est bien enregistré:

```
curl -X GET 'localhost:9200/articles/_doc/1/_source?pretty'Copier
```

Résultat :

```
{  
  "created_at" : "2021-05-07T16:24:32.000Z",  
  "title" : "mon second article sur Elasticsearch",  
  "content" : "ceci est le contenu de mon second article sur  
Elasticsearch",  
  "category" : "elasticsearch",  
  "author" : "Hatim",  
  "is_private" : false  
}Copier
```

Suppression des données dans Elasticsearch

Pour supprimer un Document d'Elasticsearch, c'est aussi simple que de rentrer des données dans Elasticsearch. La méthode HTTP à utiliser cette fois est sans surprise la méthode DELETE avec l'id du Document à supprimer:

```
curl -X DELETE 'localhost:9200/articles/_doc/1?pretty'Copier
```

Résultat :

```
{
```

```
"_index" : "articles",
"_type" : "_doc",
"_id" : "1",
"_version" : 8,
"result" : "deleted",
"_shards" : {
  "total" : 2,
  "successful" : 1,
  "failed" : 0
},
"_seq_no" : 10,
"_primary_term" : 1
}Copier
```

Si on tente de refaire une recherche sur notre Document nous obtiendrons une erreur "404 HTTP Not Found":

```
curl -X GET 'localhost:9200/articles/_doc/1/_source?pretty'Copier
```

Copier

Erreur :

```
{
  "error" : {
    "root_cause" : [
      {
```

```

    "type" : "resource_not_found_exception",
    "reason" : "Document not found [articles]/[_doc]/[1]"
  }
],
  "type" : "resource_not_found_exception",
  "reason" : "Document not found [articles]/[_doc]/[1]"
},
"status" : 404
}

```

Pour supprimer un index, il suffit tout simplement d'utiliser une nouvelle fois la requête DELETE avec comme point de terminaison le nom de l'index à supprimer:

```
curl -X DELETE 'localhost:9200/articles?pretty'Copier
```

Résultat :

```

{
  "acknowledged" : true
}Copier

```

Vérifions si notre index a bien été supprimé en vérifiant la liste des index:

```
curl -X GET 'localhost:9200/_cat/indices?pretty'Copier
```

Résultat :

```

green open .kibana_7.13.3_001
_QiHUUwqQEW3EB5YgEa1zw 1 0 4561 30 3.4mb 3.4mb

yellow open metricbeat-7.13.3-2021.07.13-000001 jHgARIOyRd-
SfFfrWzMGpw 1 1 63684 0 29.7mb 29.7mb

```

```

green open .apm-custom-link
wypmzo85R8uQ2UTcbnyxwQ 1 0 0 208b 208b

green open .kibana-event-log-7.13.3-000001
oS0Qf0ChTTiSiGzPE1aq5Q 1 0 8 43.3kb 43.3kb

yellow open apache-2021.07.12
rk1U3br_TRyUenUu6U3plw 1 1 85 124.4kb 124.4kb

green open .apm-agent-configuration
JMLcxwD8QqKj_7ZgQcKSdg 1 0 0 208b 208b

green open .async-search
6wnbxfSSREm4S1BkTIg09Q 1 0 174 100 1.4mb 1.4mb

yellow open filebeat-7.13.3-2021.07.12-000001
SSm52h8kTeeXcy4pOUM3Jg 1 1 936 0 269.5kb 269.5kb

yellow open packetbeat-7.13.3-2021.07.14-000001 AlCvCInWRM6-
B3mXUx4mVw 1 1 2770319 0 922mb 922mb

green open .tasks nmDN_yhTSti1Q-
s2kdQ2aQ 1 0 12 0 69.4kb 69.4kb

green open .kibana_task_manager_7.13.3_001
ynWJoHQ4TIWrm5W_iy8RIQ 1 0 10 29063 3.4mb 3.4mb

```

Bim, on ne le voit plus !

Conclusion

Ce chapitre avait pour but d'aider les débutants avec Elasticsearch et ne fournit que les étapes de base des opérations qu'on peut effectuer dans Elasticsearch. On peut cependant, commencer à répondre à notre question de départ "qu'est-ce qu'Elasticsearch ?"

Dans cet article, nous avons tenté de répondre à cette question à travers la compréhension de son fonctionnement et de son utilisation et nous n'avons qu'effleuré la surface pour apprendre tout ce qu'il y a à ce sujet. Mais sur la base de ce que nous avons couvert, nous pouvons résumer brièvement qu'Elasticsearch est à la base un moteur de recherche, dont l'architecture et les composants sous-jacents le rendent rapide et évolutif, au cœur d'un écosystème d'outils

complémentaires qui ensemble peuvent être utilisés pour de nombreux cas d'utilisation, notamment la recherche, l'analyse, le traitement et le stockage des données.