COMPRENDRE ET UTILISER METRICBEAT DANS LA STACK ELK

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 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 les 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 -y
```

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-https
```

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 -
```

Sous la famille redhat:

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

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 /e
```

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-md
```

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 metricbeat
```

Sous la famille redhat:

```
sudo yum install metricbeat
```

Pour exécuter Metricbeat, utilisez la commande suivante:

```
sudo systemctl start metricbeat
```

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 metricbeat
```

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/*.yml
  reload.enabled: true
```

```
  reload.period: 15s
```

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 `*.yml`. 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 metricbeat
```

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 kibana
```

Résultat :

```
? elasticsearch.service - Elasticsearch
  Loaded: loaded (/lib/systemd/system/elasticsearch.service; disabled; vendor preset
  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: enable
  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 **/etc/metricbeat/modules.d/system.yml** :

```
- 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

...
```

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 system
```

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

```
sudo metricbeat modules list
```

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):

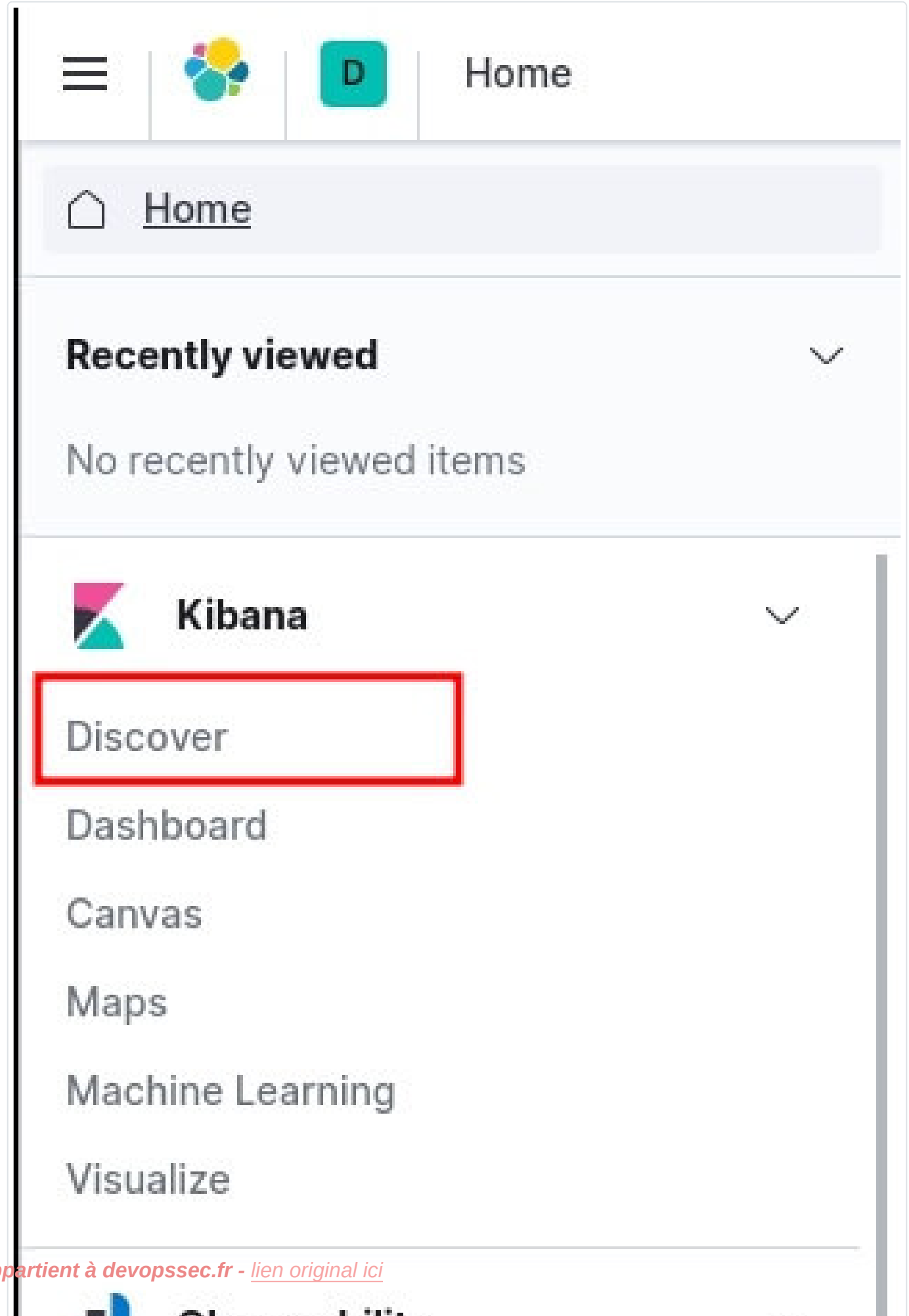
```
sudo metricbeat setup
```

Résultat :

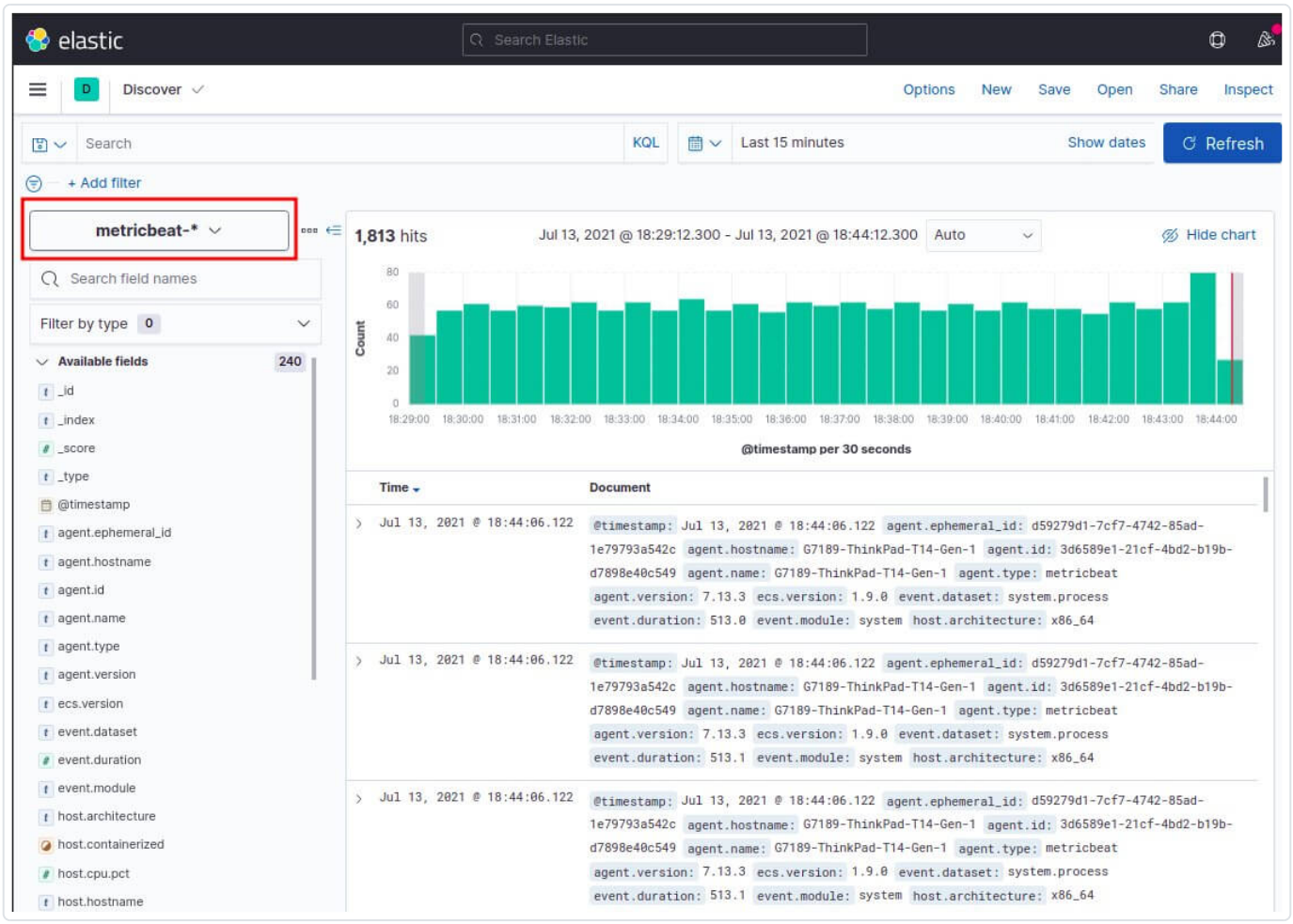
```
Overwriting ILM policy is disabled. Set `setup.ilm.overwrite: true` for enabling.

Index setup finished.
Loading dashboards (Kibana must be running and reachable)
Loaded dashboards
```

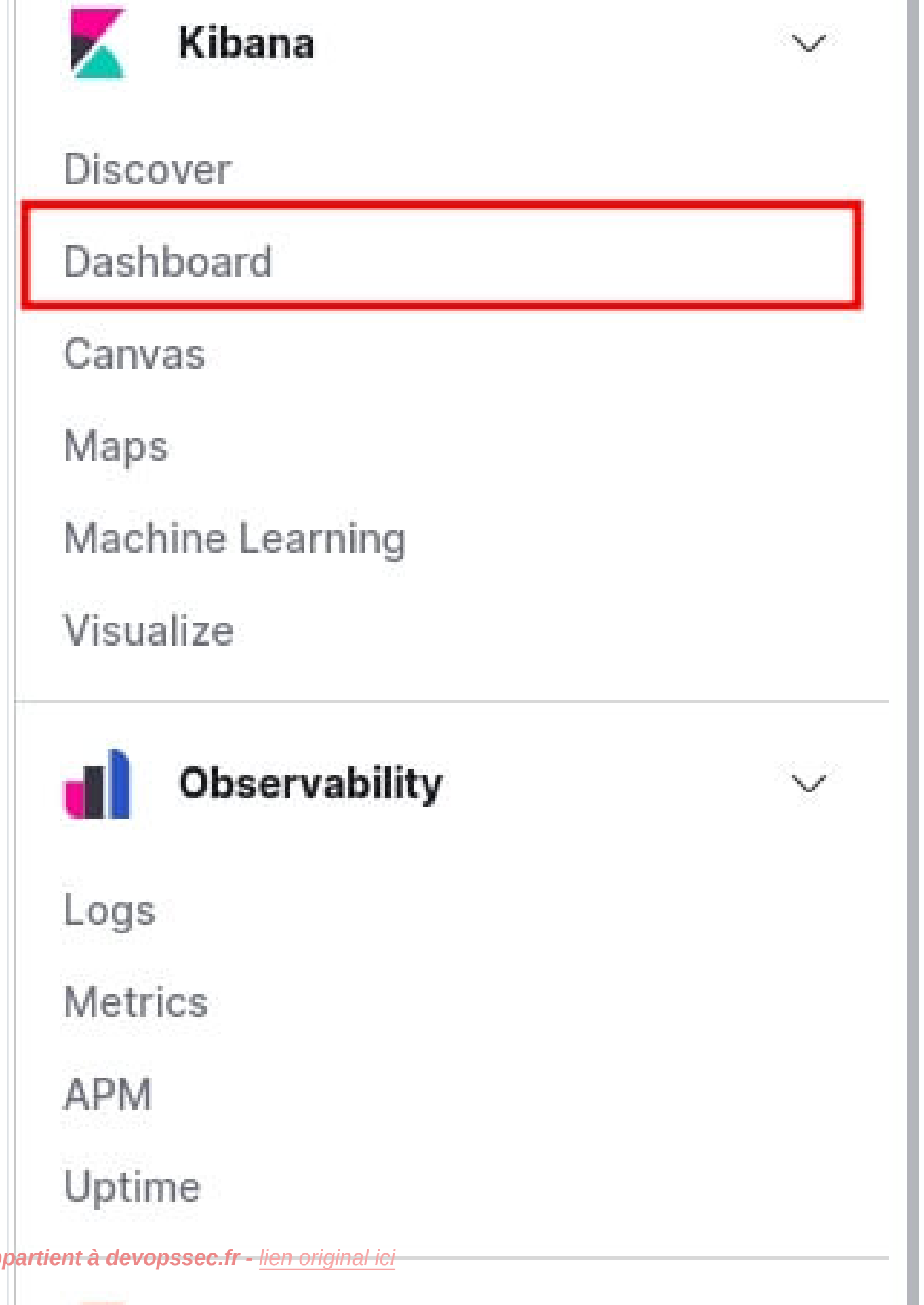
Comme pour Filebeat pour découvrir vos logs, rendez-vous dans http://localhost:5601/ et sur le menu à gauche cliquez sur Discover:



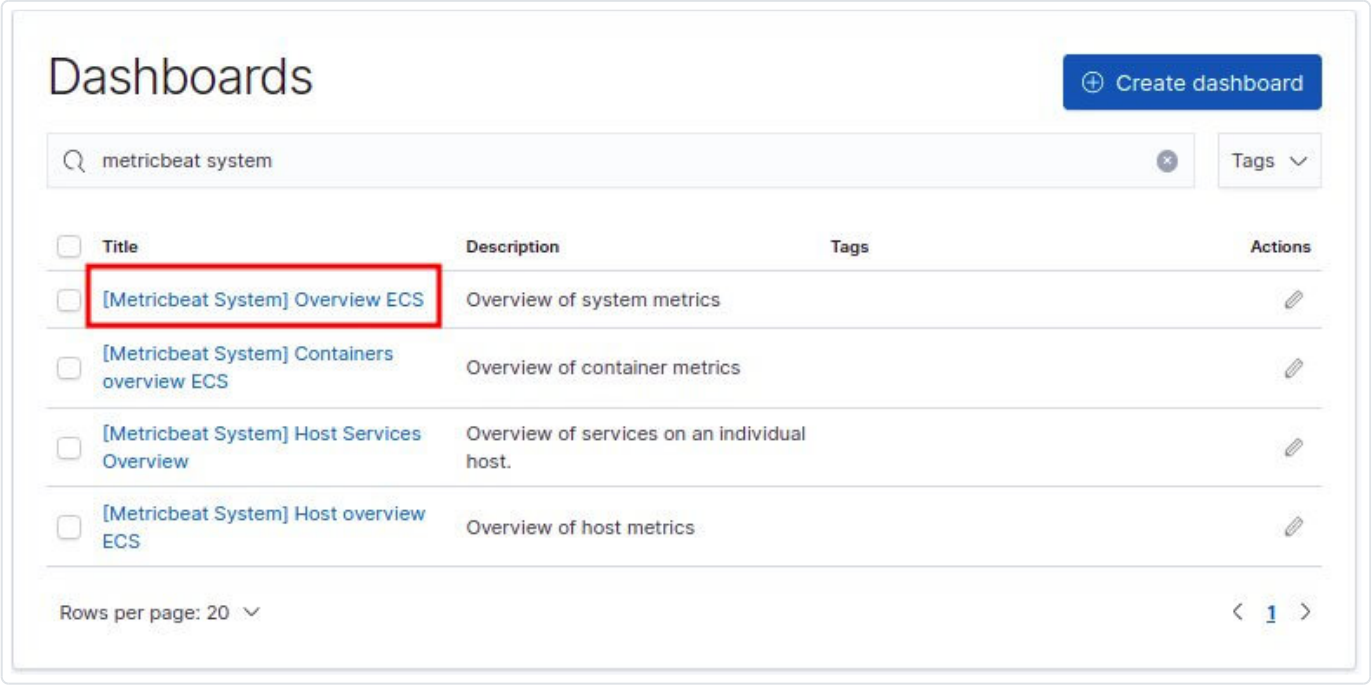
Choisissez ensuite le pattern index `metricbeat-*` pour visualiser les logs du module system Metricbeat:



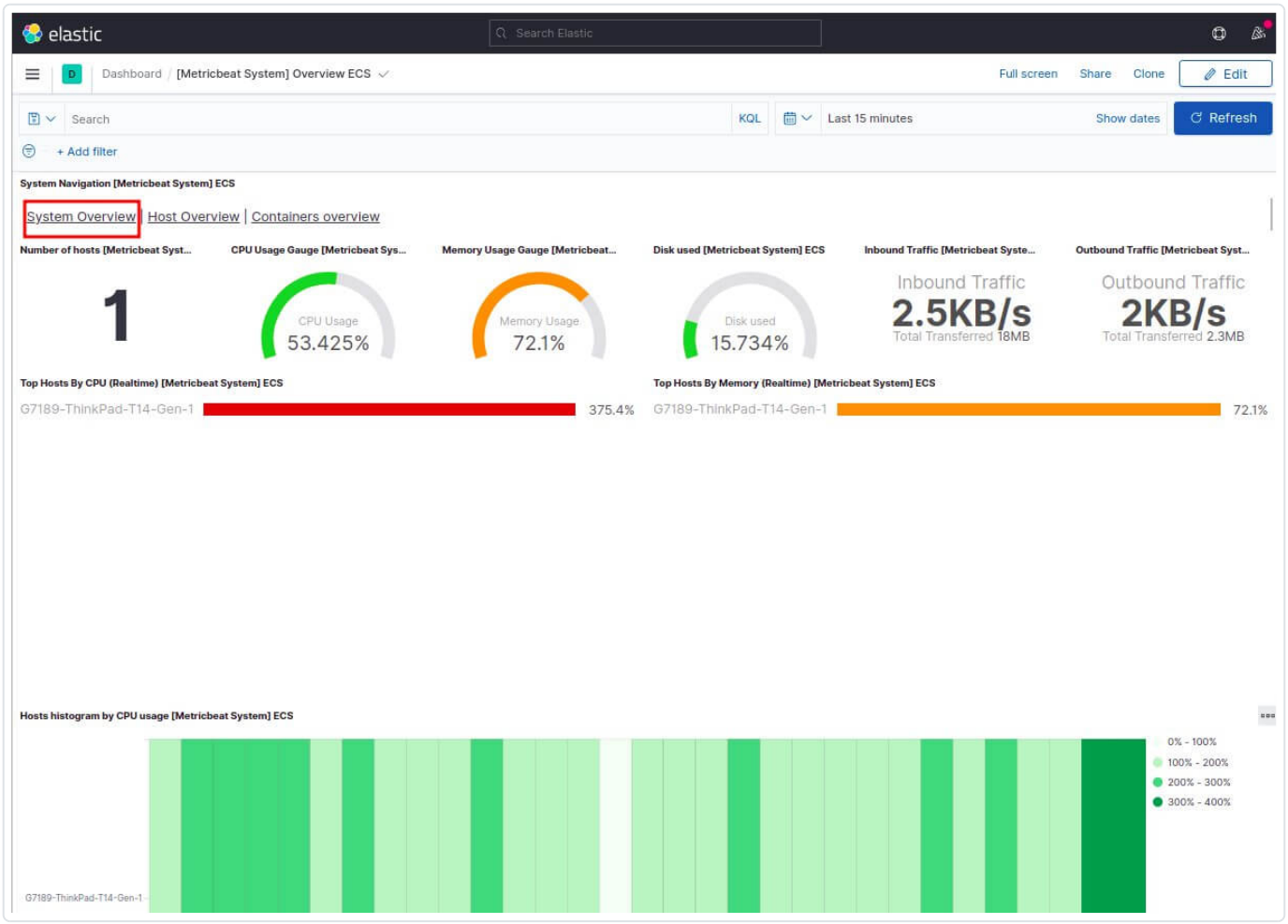
L'étape suivante est de visualiser notre tableau de bord afin de visualiser la collection de visualisations en temps réel issue par notre module system Metricbeat. Pour ce faire, sur le menu à gauche cliquez sur Dashboard:



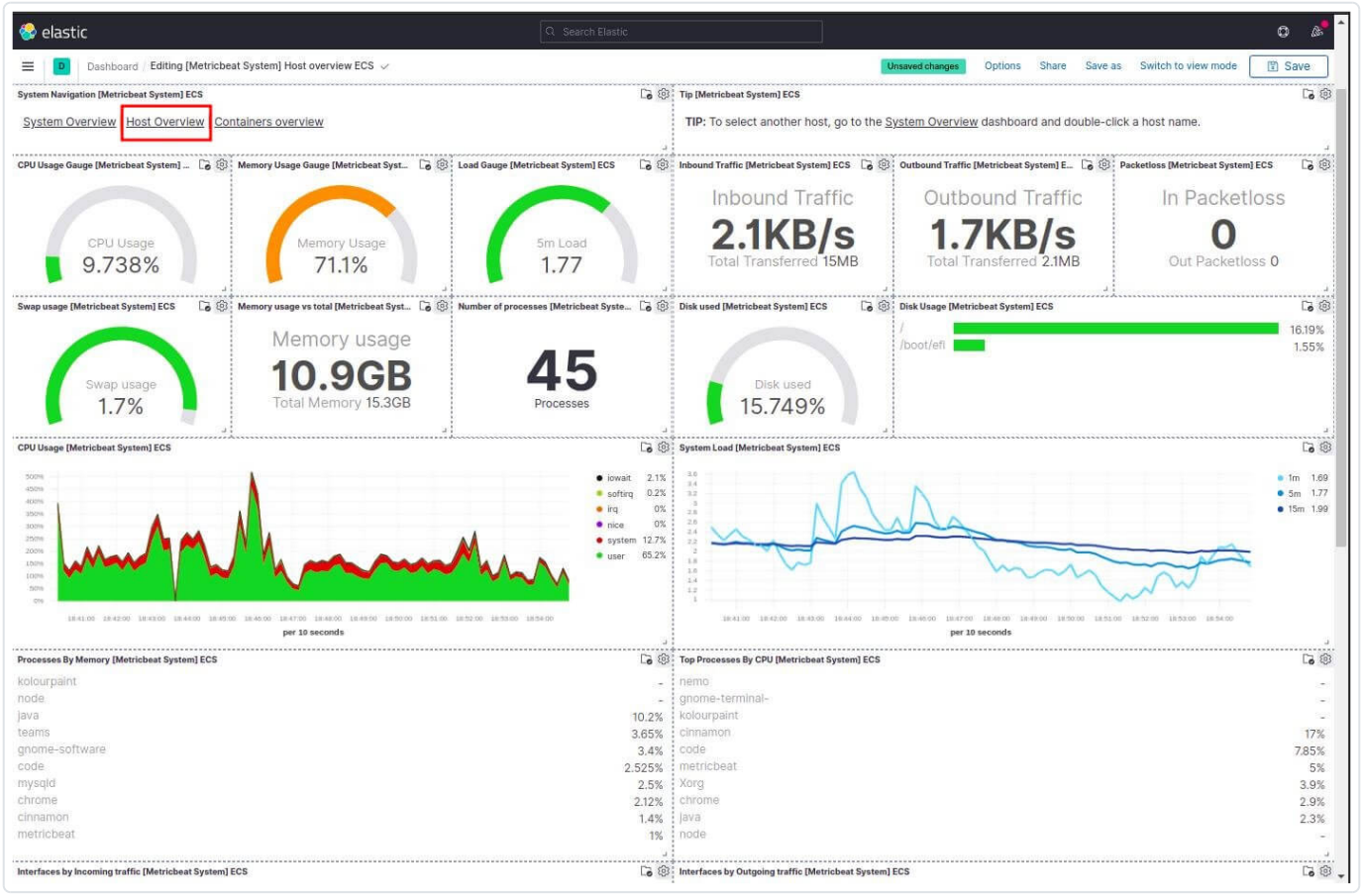
Recherchez et cliquez sur les Dashboards du module System:



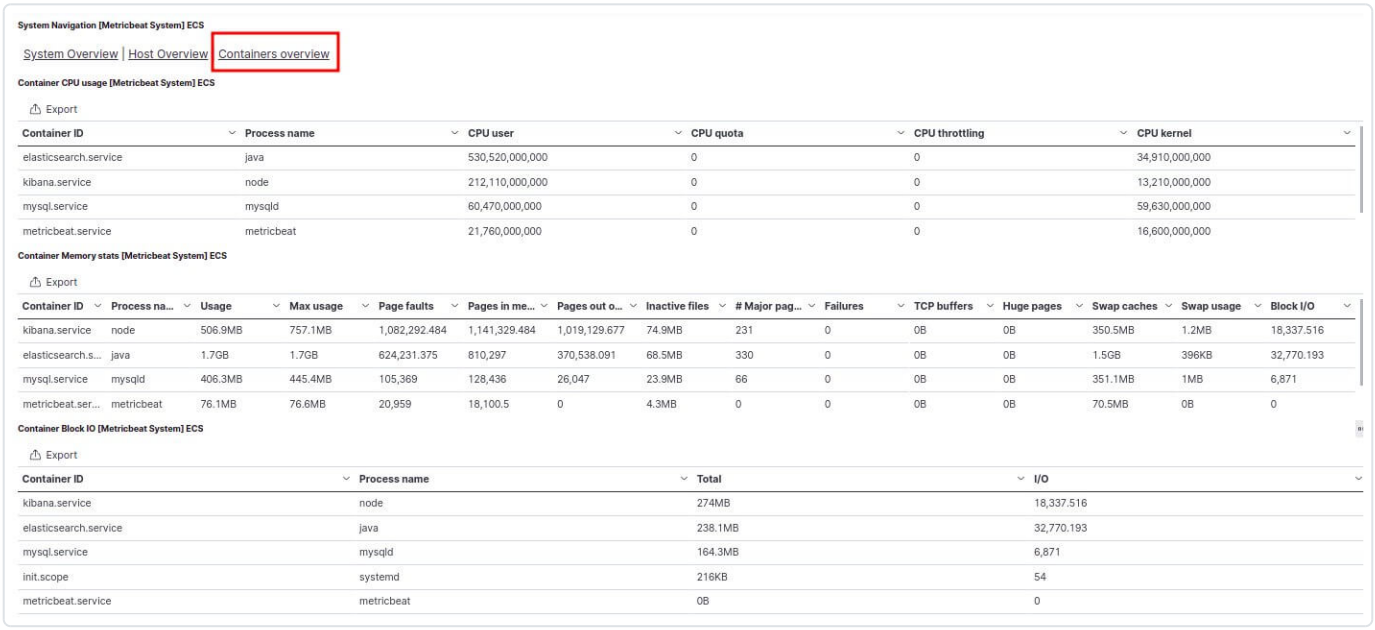
Vous obtenez ainsi 3 types de dashboard. Le premier type nommé "System Overview" ressemble à l'image ci-dessous et vous donne une moyenne la consommation de votre/vos serveur(s) concernant leur cpu, mémoire, réseaux, disque ainsi que le nombre de machines utilisant ce même module system Metricbeat:



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:



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:



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-meticbeat nginx
```

et

```
docker run -d -p 9001:80 --name httpd-meticbeat httpd
```

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

```
docker ps
```

Résultat :

```
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PO
41724bbff68d   httpd     "httpd-foreground"       7 seconds ago    Up 6 seconds    0
83f78bc139ab   nginx     "/docker-entrypoint.…"   14 seconds ago   Up 13 seconds   0
```

Ensuite, nous activons manuellement le module Docker:

```
sudo metricbeat modules enable docker
```

Maintenant que le module est activé, modifions sa configuration. Voici à quoi cela ressemble mon fichier **/etc/metricbeat/modules.d/docker.yml** :

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

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.

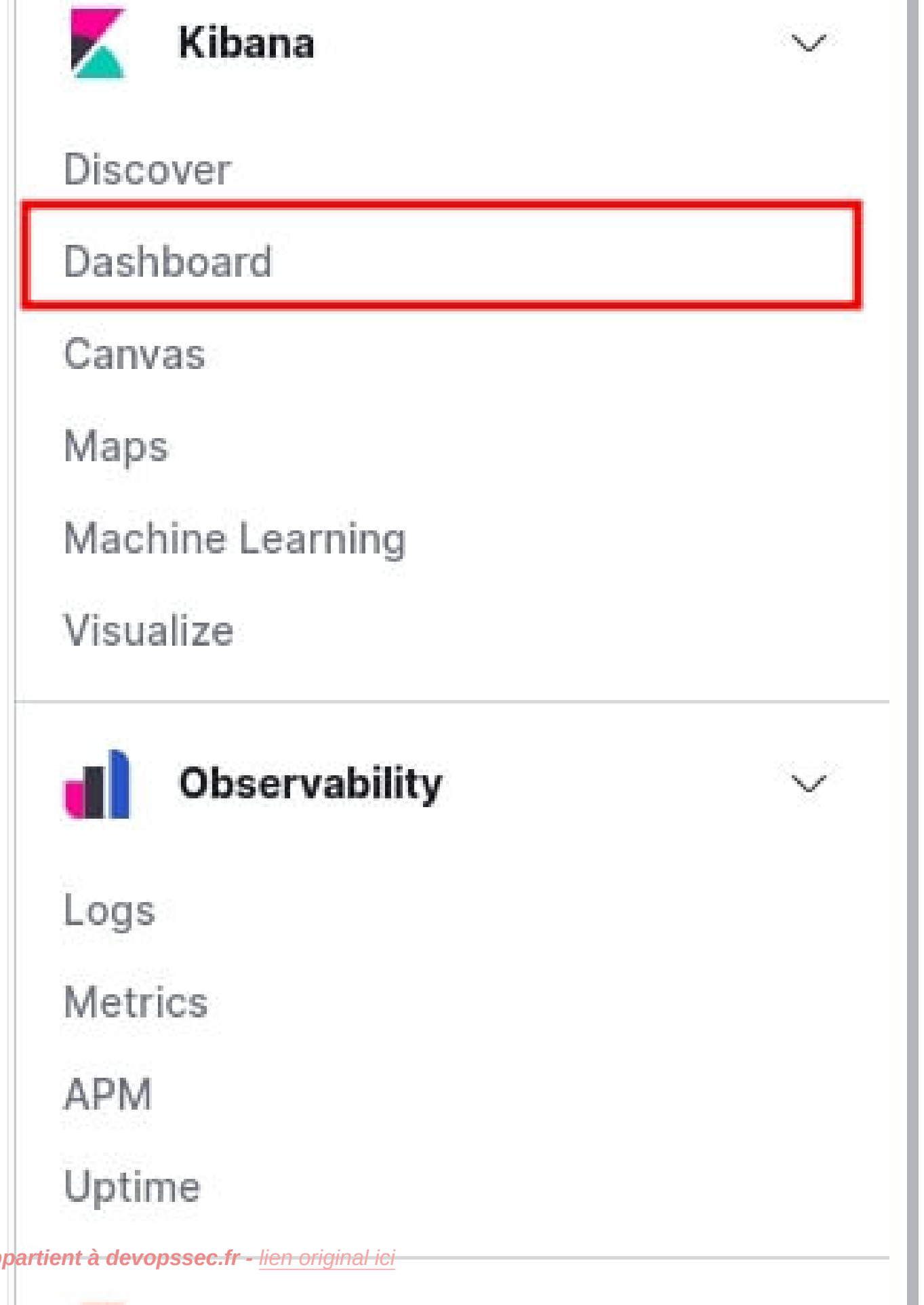
```
sudo metricbeat setup
```

Résultat :

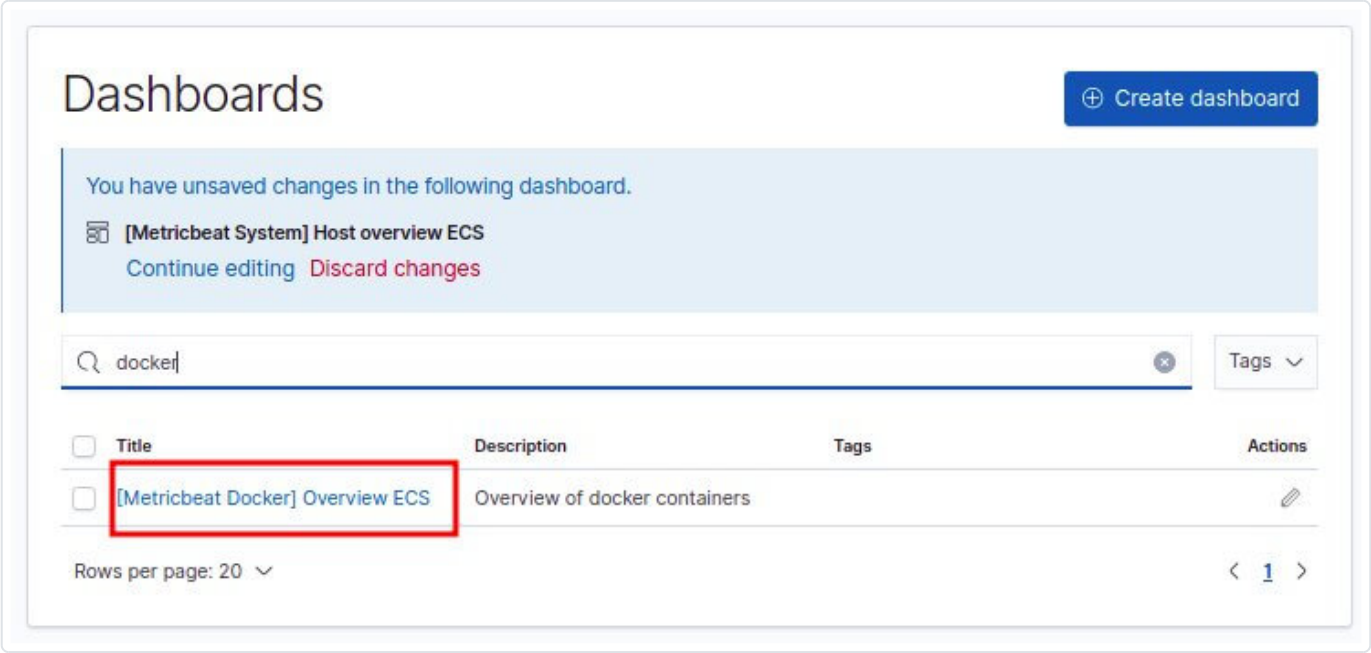
```
Overwriting ILM policy is disabled. Set `setup.ilm.overwrite: true` for enabling.

Index setup finished.
Loading dashboards (Kibana must be running and reachable)
Loaded dashboards
```

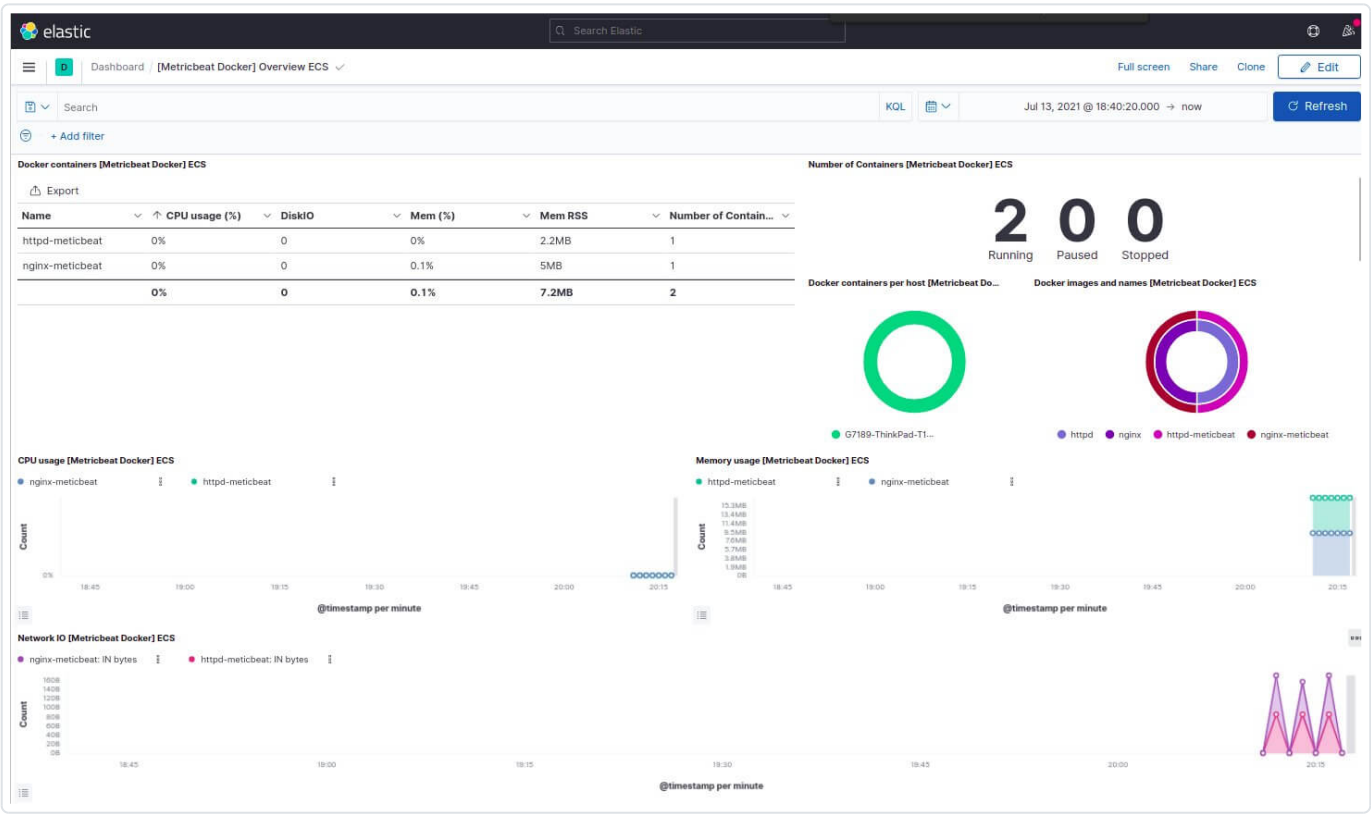
Comme pour l'exemple précédent, rendez-vous dans http://localhost:5601/ et sur le menu à gauche cliquez sur Dashboard:



Recherchez et cliquez sur les Dashboards du module Docker:



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.