

Lab 02 - Cybersecurity 3

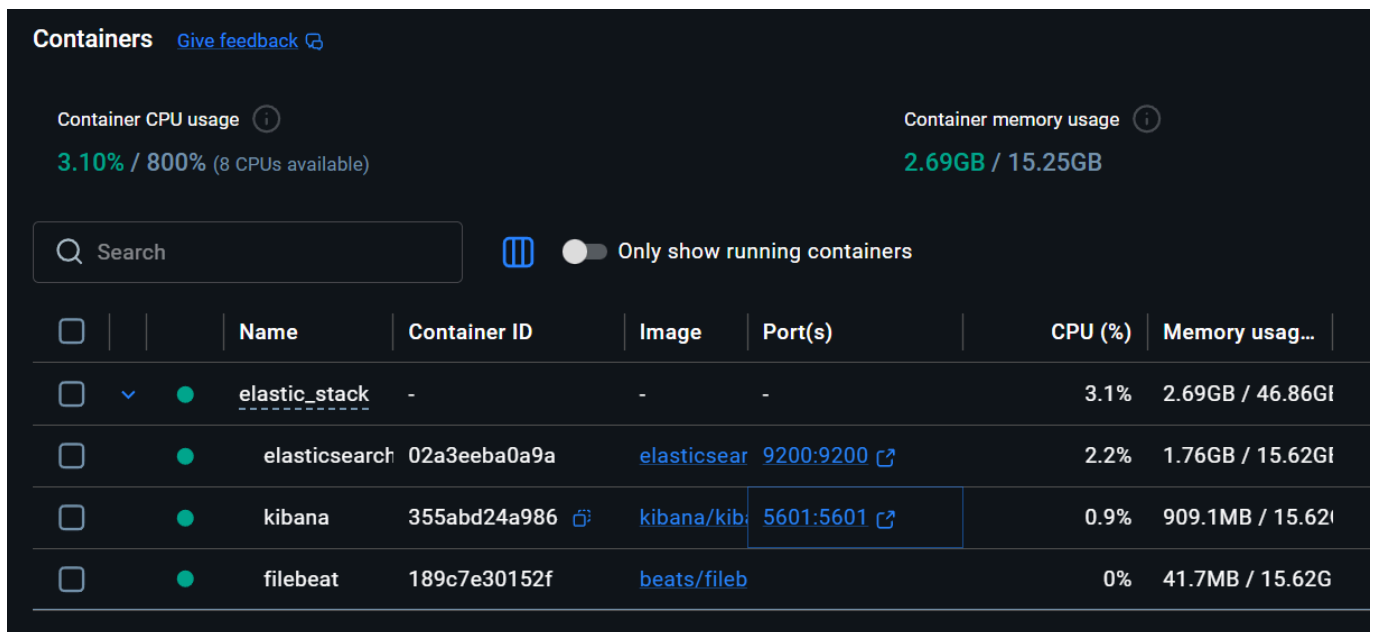
Installation Wazuh

1. Mise en place

Le `docker-compose.yml` a été modifié pour répondre aux dernières exigences Elasticsearch, incluant version, arguments et commandes d'amorçage.

`filebeat.yml` a également été modifié pour la version 9.4.4.

2. Démarrage de la stack



The screenshot shows the Docker Desktop interface with the following details:

- Containers** section with a "Give feedback" link.
- Container CPU usage:** 3.10% / 800% (8 CPUs available)
- Container memory usage:** 2.69GB / 15.25GB
- A search bar and a toggle for "Only show running containers".
- A table of running containers:

	Name	Container ID	Image	Port(s)	CPU (%)	Memory usag...
☐	elastic_stack	-	-	-	3.1%	2.69GB / 46.86Gi
☐	elasticsearch	02a3eeba0a9a	elasticsear	9200:9200	2.2%	1.76GB / 15.62Gi
☐	kibana	355abd24a986	kibana/kibana	5601:5601	0.9%	909.1MB / 15.62Gi
☐	filebeat	189c7e30152f	beats/filebeat		0%	41.7MB / 15.62Gi

3. Ingestion des logs

Les fichiers sont bien placés dans le dossier local `./data` et montés en *Read-Only* dans le conteneur `filebeat`. Une rapide inspection dans le conteneur permet de retrouver les documents :

```
> docker exec -it filebeat bash

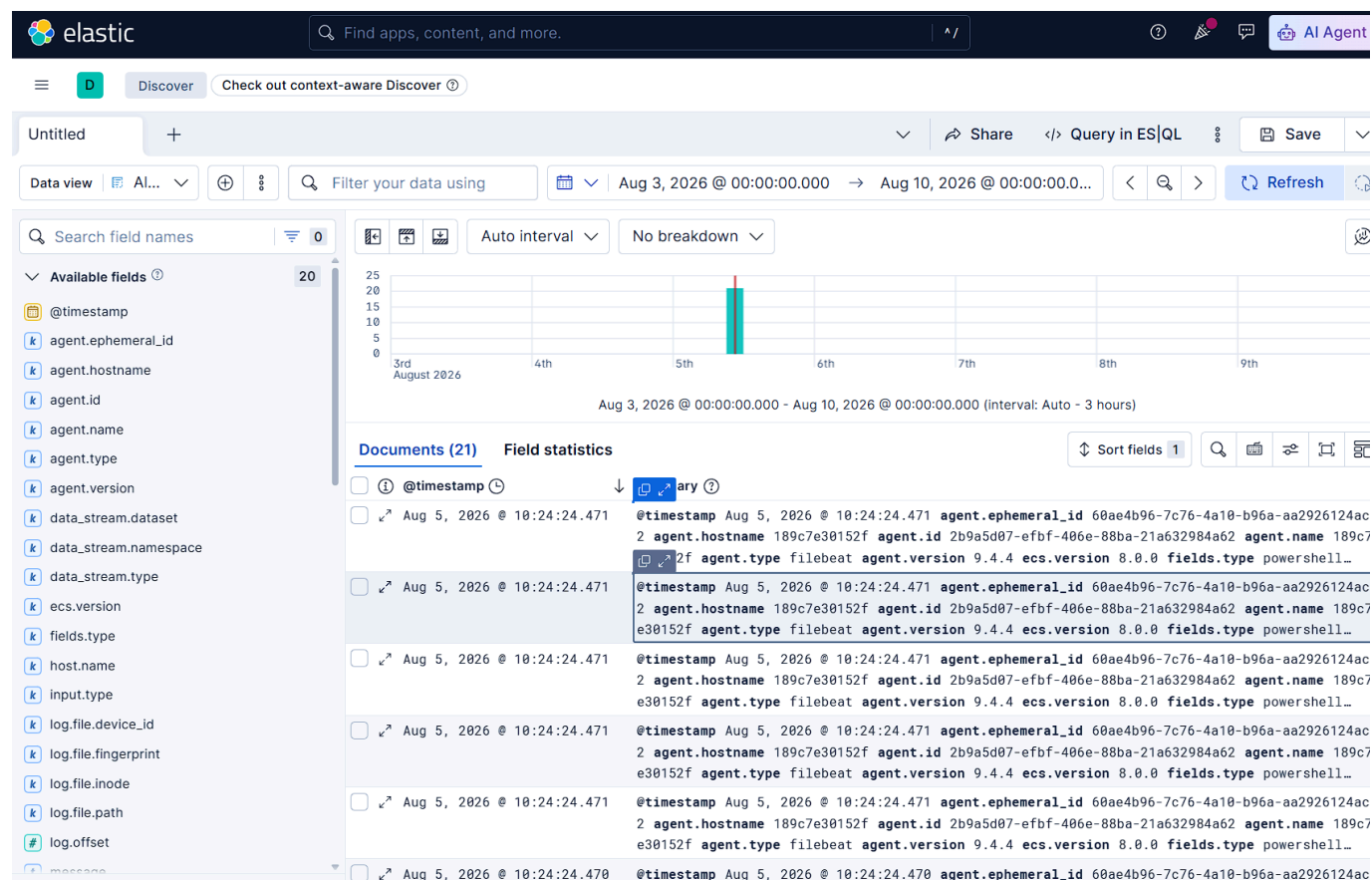
[root@189c7e30152f filebeat]# ls
LICENSE.txt  README.md  fields.yml  filebeat.reference.yml  kibana  module
NOTICE.txt  data       filebeat    filebeat.yml            logs    modules.d

[root@189c7e30152f filebeat]# pwd
/usr/share/filebeat

[root@189c7e30152f filebeat]# ls /var/log
btmptmp  hawkey.log  lastlog  powershell_eventlog.csv  private  webserver.log  wtmp
```

4. Requêtes de détection

On constate que les fichiers `filebeat-*` sont correctement journalisés dans Elasticsearch, avec du détail sous format JSON :



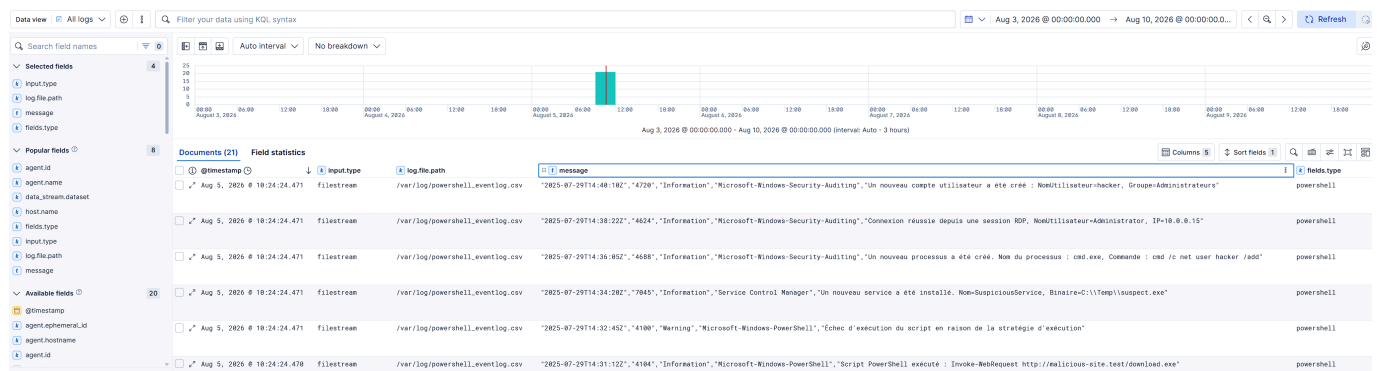
```
{
  "@timestamp": [
    "2026-08-05T08:24:24.471Z"
  ],
  "agent.ephemeral_id": [
    "60ae4b96-7c76-4a10-b96a-aa2926124ac2"
  ],
  "agent.hostname": [
    "189c7e30152f"
  ],
  "agent.id": [
    "2b9a5d07-efbf-406e-88ba-21a632984a62"
  ],
  "agent.name": [
    "189c7e30152f"
  ],
  "agent.type": [
    "filebeat"
  ],
  "agent.version": [
    "9.4.4"
  ],
  "ecs.version": [
    "8.0.0"
  ],
}
```

```

"fields.type": [
  "powershell"
],
"host.name": [
  "189c7e30152f"
],
"input.type": [
  "filestream"
],
"log.file.device_id": [
  "74"
],
"log.file.fingerprint": [
  "0552875f2772591c83e87ce749d75a90e9aa8aae5cc637f79755d3154df4ac37"
],
"log.file.inode": [
  "12103423998720744"
],
"log.file.path": [
  "/var/log/powershell_eventlog.csv"
],
"log.offset": [
  1661
],
"message": [
  "\"2025-07-29T14:40:10Z\", \"4720\", \"Information\", \"Microsoft-Windows-Security-Auditing\", \"Un nouveau compte utilisateur a été créé : NomUtilisateur=hacker, Groupe=Administrateurs\""
],
"_id": "ZOMG0Z8BjDC_E0hYQKKW",
"_index": ".ds-filebeat-9.4.4-2026.08.05-000001",
"_score": null
}

```

Je mets à profit mon écran large et je fais un tri par champs pour avoir un détail pertinent sur les logs :



A cette occasion, je corrige les requêtes pour refléter les nouvelles normes utilisées sur Elasticsearch / OpenSearch, et un référent que j'utilise qui est Datadog :

```

-- type:"web" AND (url:* OR query:*) AND message:"UNION SELECT"
fields.type:web AND message:( "UNION SELECT" OR "union select" OR "'1'='1' )

```

```
-- type:"powershell" AND (message:*"New-Object Net.WebClient"* OR
message:*"Invoke-Expression"*)
fields.type:powershell AND message:("New-Object Net.WebClient" OR "Invoke-")

-- type:"web" AND status:401
fields.type:web AND message:(" 401 ")
```

6. Création du Dashboard

Créer des métriques

Le seul champ reconnaissable par les logs est "message", qui contient les informations. En revanche, Elasticsearch ne gère "pas" nativement le CONTENU des messages en les compartimentant. Il faut pouvoir extraire ces données. La solution la plus efficace est de passer une instruction dans Filebeat, mais dans un vrai environnement de remontée de logs, on n'a que rarement accès à l'agent émetteur. Aussi va-t-on essayer de faire des métriques dans Kibana directement.

Pour ça, je vais dans **Dev Tools > Console** et je fais, aidé de Kimi K2.7, un "Ingest Pipeline Conditionnel" :

```
PUT _ingest/pipeline/multi_log_parser
{
  "description": "Détection et sépare les logs Web et PowerShell/Windows",
  "processors": [
    {
      "dissect": {
        "description": "Parse les logs Web au format Apache/Nginx",
        "if": "ctx.message != null && ctx.message.contains('HTTP/')",
        "field": "message",
        "pattern": "%{web_client_ip} - - [%{web_timestamp}] \"%{web_http_method} %
{web_request_path} HTTP/%{web_http_version}\" %{web_status_code} %{web_bytes} \"%
{web_referrer}\" \"%{web_user_agent}\""
      }
    },
    {
      "csv": {
        "description": "Parse les logs Windows/PowerShell au format CSV",
        "if": "ctx.message != null && (ctx.message.contains('Microsoft-Windows')
|| ctx.message.startsWith('\"202'))",
        "field": "message",
        "target_fields": [
          "pws_timestamp",
          "pws_event_id",
          "pws_level",
          "pws_provider",
          "pws_message"
        ]
      }
    }
  ]
}
```

The screenshot shows the Elastic Dev Tools console. The 'Shell' tab is active, displaying a PUT request to create an ingest pipeline named 'multi_log_parser'. The request body is a JSON object with a description in French and two processors: 'dissect' and 'csv'. The 'dissect' processor is configured to parse web logs, and the 'csv' processor is configured to parse Windows/PowerShell logs. The response on the right shows the pipeline was created successfully with 'acknowledged': true. The status bar at the bottom indicates a 200 OK response in 169 ms.

```

10 PUT /my-index
11
12 PUT _ingest/pipeline/multi_log_parser
13 {
14   "description": "Détecte et sépare les logs Web et
15     PowerShell/Windows",
16   "processors": [
17     {
18       "dissect": {
19         "description": "Parse les logs Web au format Apache/
20           Nginx",
21         "if": "ctx.message != null && ctx.message.contains
22           ('HTTP/')",
23         "field": "message",
24         "pattern": "%{web_client_ip} - - [%{web_timestamp}]
25           \\\"{web_http_method} %{web_request_path} HTTP/%
26           {web_http_version}\\\" %{web_status_code} %
27           {web_bytes} \\\"{web_referrer}\\\" \\\"%
28           {web_user_agent}\\\"\""
29       }
30     },
31     {
32       "csv": {
33         "description": "Parse les logs Windows/PowerShell au
34           format CSV",
35         "if": "ctx.message != null && (ctx.message.contains
36           ('Microsoft-Windows') || ctx.message.startsWith
37           ('Microsoft-Windows'))"
38       }
39     }
40   ]
41 }
  
```

Click to send request

```

1 {
2   "acknowledged": true
3 }
  
```

200 - OK 169 ms

Pas de succès. Je tente alors avec **Stack Management > Data Views > Logs** de faire des champs personnalisés :

```

if (params._source.message != null) {
  def msg = params._source.message;
  if (msg instanceof List && !msg.isEmpty()) { msg = msg[0]; }

  // Exécute uniquement si le message ressemble au log Windows
  if (msg instanceof String && msg.contains('Microsoft-Windows')) {
    String[] parts = msg.splitOnToken('"', '"');
    if (parts.length > 1) {
      emit(parts[1].replace('"', ' ').trim()); // Extrait l'ID d'évènement
    }
  }
}
  
```

```

if (params._source.message != null) {
  def msg = params._source.message;
  if (msg instanceof List && !msg.isEmpty()) { msg = msg[0]; }

  // Exécute uniquement si le message contient une requête HTTP
  if (msg.contains('HTTP')) {
    emit(msg);
  }
}
  
```

```

if (msg instanceof String && msg.contains('HTTP/')) {
    String[] parts = msg.splitOnToken(' ');
    if (parts.length >= 9) {
        emit(parts[8].trim()); // Extrait le code de réponse (200, 401, 403...)
    }
}
}
}

```

J'ai des résultats :

Keyword

×

Set custom label

Create a label to display in place of the field name in Discover, Maps, Lens, Visualize, and TSVB. Useful for shortening a long field name. Queries and filters use the original field name.

×

Set custom description

Add a description to the field. It's displayed next to the field on the Discover, Lens, and Data View Management pages.

✓

Set value

Set a value for the field instead of retrieving it from the field with the same name in `_source`.

Define script

```

1  if (params._source.message != null) {
2      def msg = params._source.message;
3      if (msg instanceof List && !msg.isEmpty()) { msg
        = msg[0]; }
4      |
5      if (msg instanceof String && msg.contains
        ('Microsoft-Windows')) {
6          String[] parts = msg.splitOnToken(' ','');
7          if (parts.length > 1) {
8              emit(parts[1].replace('\"', ' ').trim());
          }
        }
      }
    }

```

Runtime fields without a script retrieve values from `_source`. If the field doesn't exist in `_source`, a search request returns no value. [Learn about script syntax.](#)

Preview

From: filebeat-*

Document ID

<

>

Filter fields

4625

pws_timestamp	4625
agent.version	9.4.4
ecs.version	8.0.0
fields.type	powershell
host.name	189c7e30152f
input.type	filestream
log.file.device_id	74
log.file.fingerprint	0552875f2772591c83e...
log.file.inode	12103423998720744
log.file.path	/var/log/powershell_eve...
log.offset	258
message	"2025-07-29T14:25:12...

Show less

Logs

Edit data view

Index pattern: filebeat-*

Time field: @timestamp

★ Default

Fields (6698)

Field filters (0)

Relationships (0)

Search

Field type16

Schema type1

Refresh

Add field

Name ↑	Type ↓	Format ↓	Searchable ↓	Aggregat... ↓	Excluded ↓	Actions
k pws_event_id	keyword		●	●		
k pws_level	keyword		●	●		
k pws_message	keyword		●	●		
k pws_provider	keyword		●	●		
k pws_timestamp	keyword		●	●		
k web_client_ip	keyword		●	●		
k web_http_method	keyword		●	●		
k web_request_path	keyword		●	●		
k web_status_code	keyword		●	●		

Je peux maintenant faire un VRAI tableau avec mes métriques :

elastic

Find apps, content, and more.

AI Agent

Dashboards

Create

Explore in Discover

Inspect

Settings

Cancel

Save to library

Save and return

Data view

Logs

Filter your data using KQL s

Aug 5, 2026 @ 10:24:24...

Aug 5, 2026 @ 10:24:24.472

Refresh

Search field names

0

Records

Selected fields2

Available fields28

Count of records

Area

Stacked

Logs

Horizontal axisOptional

@timestamp

Vertical axis

Count of records

Add or drag-and-drop a field

BreakdownOptional

Add or drag-and-drop a field

Suggestions

Refresh

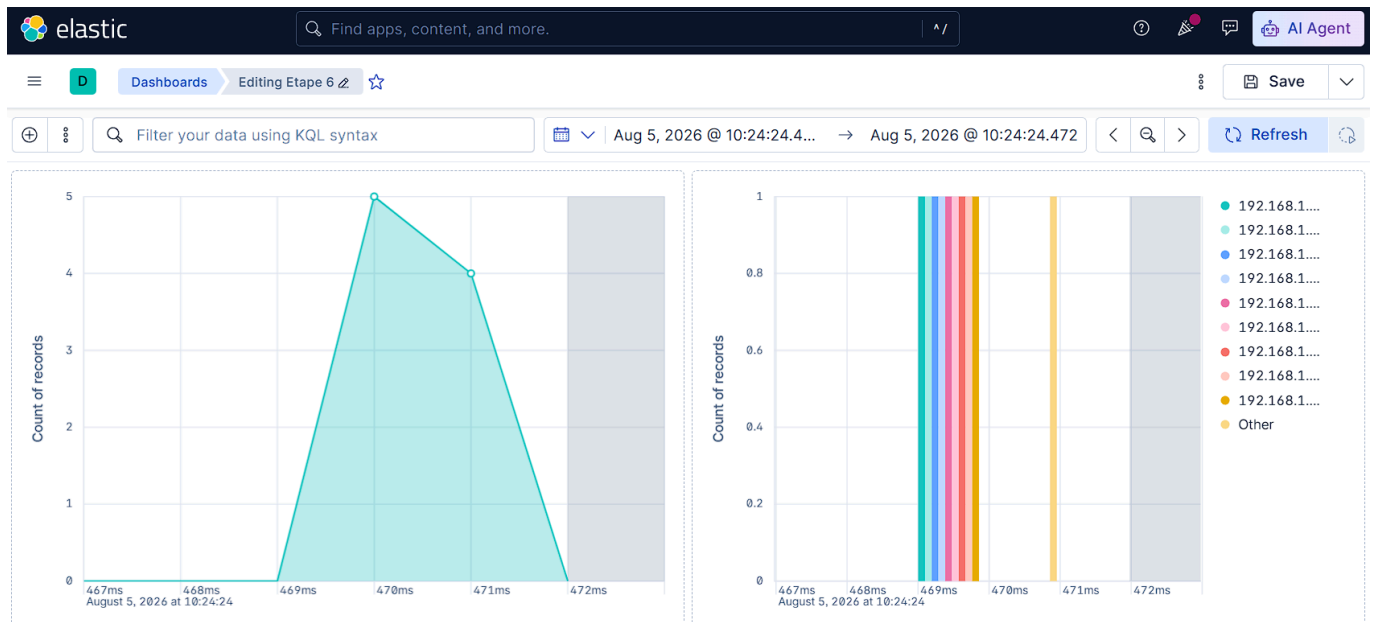
Current visualization

Count of records8

Area stacked

Bar p

Et enfin faire une visualisation :

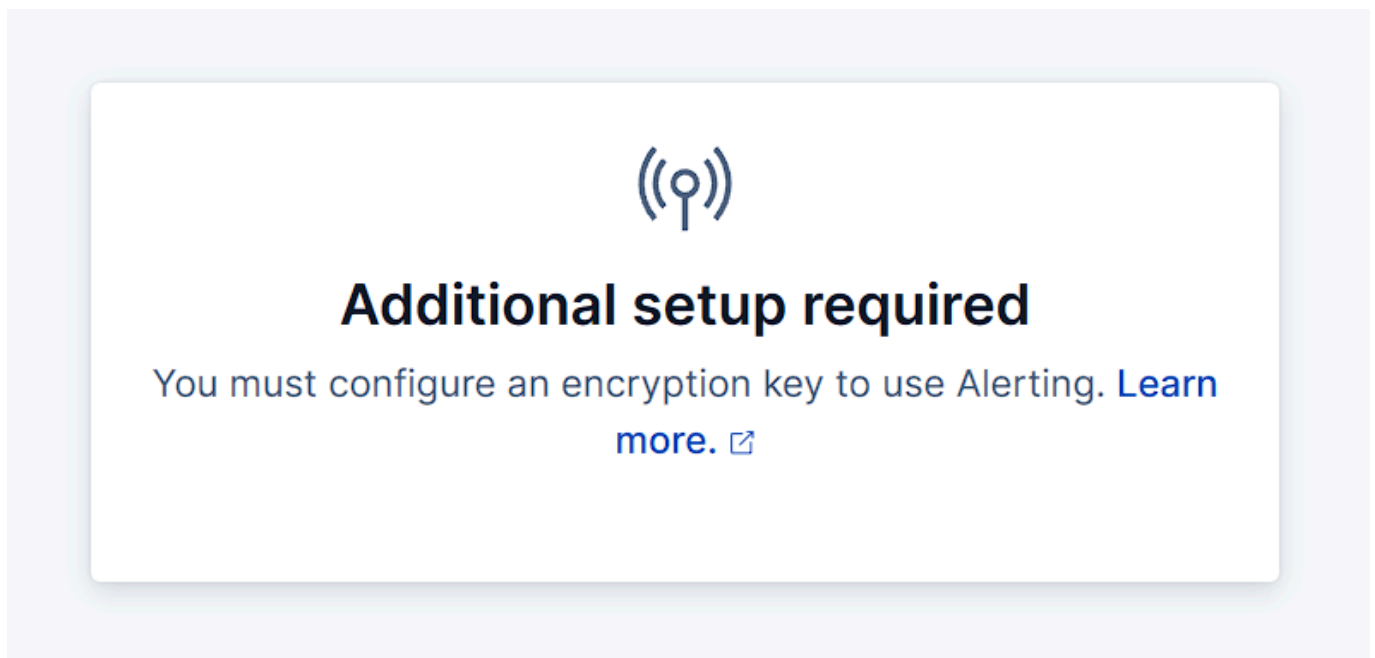


Hélas le jeu de données est

7. Alertes automatisées

Pour l'alerte demandée, je vais dans **Stack Management > Alerts and Insights > Rules** bien que ça ne soit pas très instinctif, mais pour les besoins du cours, il faudra y opérer.

Interruption



Je dois rajouter un élément au `kibana.yml` du conteneur :

```
xpack.encryptedSavedObjects.encryptionKey:
"ma_cle_secrete_super_longue_de_32_caracteres!"
```

J'édite aussi le `docker-compose.yml` pour déploiement automatique.

Je définis les conditions de déclenchement (ici, très parano, faute de gros jeux de données) :

The screenshot shows the Elastic Rules configuration page for an 'Index threshold' rule. The rule is designed to alert when an aggregated query meets a threshold. The configuration includes:

- Select indices:** INDEX filebeat-*
- Define the condition:**
 - WHEN count()
 - OVER all documents
 - IS ABOVE 5
 - FOR THE LAST 1 minute
- Filter:** A search bar for KQL expressions to limit the scope of alerts.
- Visual representation:** A line graph showing the count() over time. The y-axis is labeled 'count()' and ranges from 0 to 20. The x-axis shows time from August 5, 2026, 11:55 to 12:20. A sharp peak is visible around 12:08, reaching a count of approximately 20.

Pour les actions, je suis obligé d'activer la licence d'essai 30 jours (ça va, on a le temps) mais je n'ai que deux actions de disponibles, là où il y a normalement des dizaines de choix.

POUR LES BESOINS DE L'EXERCICE je ne peux pas générer de Workflow complet. En revanche je peux montrer un exemple défini sur Datadog, utilisant un Webhook paramétré dans Slack pour prévenir les utilisateurs d'une instance EC2 qui tourne à vide :

The screenshot shows the Datadog configuration interface for notifications and automations. The configuration includes:

- Recipient summary:** 2 Slack recipients.
- Monitor name:** `{{aws_account_name.name}}` EC2: idle individual instance running for more than 4 hours (`{{name.name}}`)
- Edit/Preview:** A section for configuring the notification message. The message template is as follows:


```

{{^is_renotify}}
{{#is_alert}}
[Redacted]
[Redacted]
@slack-ec2_instances_alerts <@{{user.name}}>
Hello, we just received an alert that your EC2 {{name.name}} instance has been running for more than 4 hours without any activity.
As a reminder, individual EC2 instances are quite expensive and must be stopped after each use.
Would it be possible to turn it off?
{{/is_alert}}
{{/is_renotify}}
      
```

J'ai déjà fait du déploiement :

- Courriel
- JIRA OpsGenie
- Webhook Slack
- Webhook Discord
- Application WhatsApp pour les urgences VIP