Member-only story

How to read files in Elasticsearch? (doc, docx, pdf)

Maciej Szymczyk · Follow
5 min read · May 27, 2020

You will probably be surprised by this information. Elasticsearch is used for ... searching. Yes. This is true. It turns out that it can also be used to index the contents of doc, docx, pdf files, etc. In this post, we'll look at how to do it, how to change the analyzer, and how to "lose" a file if we keep it on S3 or other filesystem.

What for?

Not always the phrase we are looking for is in the file name, title and other metadata provided. Imagine a portal that allows you to collect and search for scientific articles. Each article is a separate PDF file. Adding abstracts to the search area would already noticeably increase the usability of the portal. ScienceDirect uses Elasticsearch for a reason.

Environment

To enable file analysis in Elasticsearch, you need Ingest Attachment Processor Plugin. In the case of Docker, we could manually enable the terminal inside the container with the following command.

```
sudo docker exec -it container-name bash
```

This is a bad idea. If the container is removed, we will have to repeat this action. To avoid this, I slightly modified Docker Compose with ELK from the Docker entry and added a simple Dockerfile.

```
FROM docker.elastic.co/elasticsearch/elasticsearch:7.6.0
RUN bin/elasticsearch-plugin install --batch ingest-attachment
```

So docker-compose.yml looks something like this:

```
version: '2.2'
services:
elasticsearch:
build: ./custom-elasticsearch/
#    image: docker.elastic.co/elasticsearch/elasticsearch:7.6.0
restart: unless-stopped
...
```

Pipeline preparation

What is Pipeline at all? This is the definition of a series of processors that will be performed in the same order in which they were declared. In other words, the document that we throw into the base will be passed through each defined processor. We can thus enrich the document with new fields, transform and even delete it if the defined condition is met.

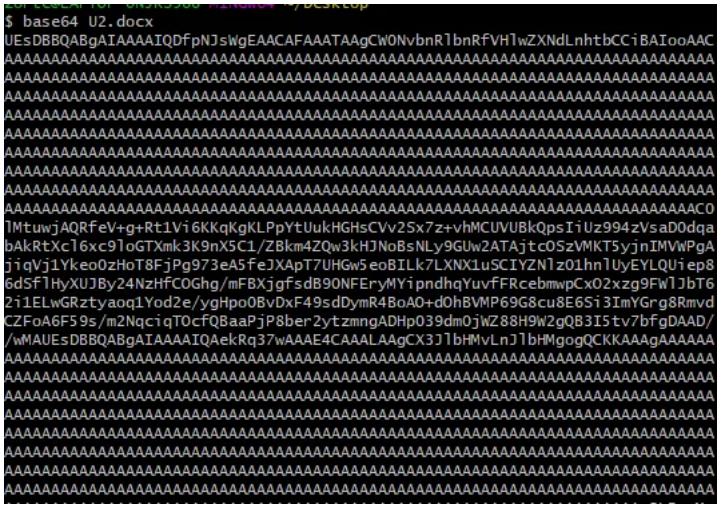
The previously added plugin contains a pipeline that will extract and analyze the added file. The files are transferred in the form of Base64. Below is a simple pipeline declaration.

```
PUT _ingest/pipeline/attachment
{
  "description": "What did you hide in this file? (¬‿¬)",
  "processors": [
    {
      "attachment": {
        "field": "data"
      }
    }
```

```
  ]
}
```

Adding a file

I prepared doc, docx, pdf files with the content of the song U2 — I Still Haven Found What I'm Looking For. One of the pdf consists of a screenshot screenshot of the text. Does the plugin have OCR in it? We will find out.



Pasting Base64 into Postman or Dev Tools in Kiban is a poor idea. It takes too much space. That's why we'll use CURL. We will add a document with the filename and data parameters to the songs index. *?pipeline=attachment* indicates the previously defined Pipeline.

```
(echo -n '{"filename":"U2.docx", "data": "'; base64 ./U2.docx; echo
'"}') |
curl -H "Content-Type: application/json" -d @-
http://192.168.114.128:9200/songs/_doc/1?pipeline=attachment
```

The result is a database document that looks like this (I allowed myself to remove base64 and part of the song):

```
{
  "_index" : "songs",
  "_type" : "_doc",
  "_id" : "1",
  "_version" : 1,
  "_seq_no" : 0,
  "_primary_term" : 1,
  "found" : true,
  "_source" : {
    "filename" : "U2.docx",
    "data" :
"UEsDBBQABgAIAAAAIQDfpNJsWgEAACAFAAATAAgCW0NvbnRlbnRfVHlwZXNdLnhtbCCi
BAIooAACAAAAA...AsyoAAGRvY1Byb3BzL2FwcC54bWxQSwUGAAAAAAsACwDBAgAAXy0A
AAAA",
    "attachment" : {
      "date" : "2020-02-21T19:14:00Z",
```

```
      "content_type" : "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
      "author" : "Maciej Szymczyk",
      "language" : "en",
      "title" : "U2 - I Still Haven't Found What I'm Looking For",
      "content" : """I have climbed highest mountain
I have run through the fields
Only to be with you
Only to be with you
...
But I still haven't found
What I'm looking for
But I still haven't found
What I'm looking for""",
      "content_length" : 931
    }
}
```

Search

Searching for the word “kissed” returns the added file to us

```
POST /songs/_search
{
  "query": {
    "query_string": {
      "default_field": "attachment.content",
      "query": "kissed"
    }
  }
}
```

But the word “kiss” doesn’t give us any results.

```
POST /songs/_search
{
  "query": {
    "query_string": {
      "default_field": "attachment.content",
      "query": "kiss"
    }
  }
}
```

This is because we did not prepare the index before adding the first record. I mean English analyzer. In addition to removing punctuation marks and hyphens, it will transform verbs to their basic form (kissed -> kiss). We can check how such an analyzer works with the following query.

```
POST /_analyze
{
  "analyzer": "english",
  "text": """I have climbed highest mountain
             I have run through the fields"""
}
```

Let's correct this error. Let's delete the index, add the index definitions and the song file.

```
DELETE /songs

PUT /songs
{
  "mappings": {
    "properties": {
      "attachment.content": {
        "type": "text",
        "analyzer": "english"
      }
    }
  }
}
```

Now using *query_string* returns a record to us regardless of whether we type kiss or kissed. Remember that in q*uery_string* / *match* query (as opposed to term query) the given phrase goes through the analyzer used in a given field. So kissed is still transformed into a kiss.

But I don't need to analyze the entire file

In order not to waste time / storage on the entire file, we can limit the content to be analyzed. The same applies to file metadata.

```
PUT _ingest/pipeline/better_attachment
{
  "description" : "What did you hide in this file? better version
(¬‿¬)",
  "processors" : [
    {
      "attachment" : {
        "field" : "data",
        "properties": [ "content", "title" ],
        "indexed_chars" : 20,
        "indexed_chars_field" : "max_size"
      }
    }
  ]
}
```

I don't want to keep files on Elasticsearch. I keep them on S3

In this case, we can add another processor, namely Remove Processor.

```
PUT _ingest/pipeline/even_better_attachment
{
  "description" : "What did you hide in this file? even better
version (¬‿¬)",
  "processors" : [
    {
      "attachment" : {
        "field" : "data",
        "properties": [ "content", "title" ],
        "indexed_chars" : 20,
        "indexed_chars_field" : "max_size"
```

```
      },
      "remove":{
        "field":"data"
      }
    }
  ]
}
```

Now the document looks like this:

```
{
  "_index" : "songs",
  "_type" : "_doc",
  "_id" : "1",
  "_version" : 1,
  "_seq_no" : 0,
  "_primary_term" : 1,
  "found" : true,
  "_source" : {
    "filename" : "U2.docx",
    "attachment" : {
      "title" : "U2 - I Still Haven't Found What I'm Looking For",
      "content" : "I have climbed highe"
    }
  }
}
```

OCR

Doesn't work :-)

Github repo

https://github.com/zorteran/wiadro-danych-elasticsearch-ingest-attachment

Elasticsearch Elastic Searching Search Big Data

Written by Maciej Szymczyk

289 Followers

Software Developer, Big Data Engineer, Blogger (https://wiadrodanych.pl), Amateur Cyclists & Triathlete, @maciej_szymczyk

More from Maciej Szymczyk

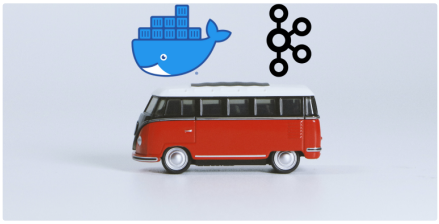
Maciej Szymczyk

Efficient SIEM and Detection Engineering in 10 steps

SIEM systems and detection engineering are not just about data and detection rules....

✦ · 8 min read · Mar 25

52 1



Maciej Szymczyk in The Startup

Dockerizing a Kafka Streams app

Docker images are easy to use. We do not need to install a specific version of the...

✦ · 2 min read · May 12, 2020

144 2

Maciej Szymczyk in ITNEXT

How to Elastic SIEM (part 1)

IT environments are becoming increasingly large, distributed and difficult to manage. All...

✦ · 6 min read · Aug 20, 2020

17

Maciej Szymczyk

How to use Variables and XCom in Apache Airflow?

It is said that Apache Airflow is CRON on steroids. It is gaining popularity among tools...

✦ · 4 min read · Dec 11, 2020

36

See all from Maciej Szymczyk

Recommended from Medium

Rishab Batra

Indexing Data into Elasticsearch using Python

Search with Elasticse

3 min read · May 16

43

Vaishnav Manoj in DataX Journal

JSON is incredibly slow: Here's What's Faster!

Unlocking the Need for Speed: Optimizing JSON Performance for Lightning-Fast Apps...

16 min read · Sep 28

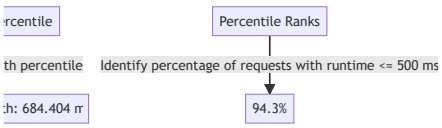
8.4K 111

Lists

New_Reading_List
174 stories · 191 saves

Natural Language Processing
837 stories · 390 saves

Staff Picks
500 stories · 445 saves



mastinder@gmail.com

Elasticsearch Queries

In this guide, we'll dive deep into some Elasticsearch queries, understand their...

2 min read · Oct 22

1

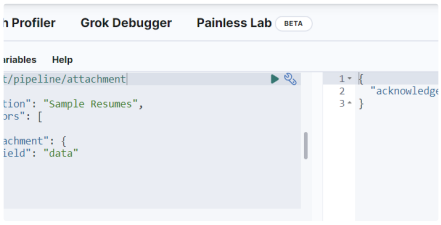
Rany ElHousieny in Level Up Coding

Integrating ChatGPT with ElasticSearch for Efficient Contex...

How to adjust ChatGPT responses to your Context

23 min read · Jul 26

55 1



Meltem YILMAZ

Elasticsearch — Attach PDF Files

Hi everyone,

3 min read · Jun 14

17

Sai Mohan Kesapragada

Snowflake to Elasticsearch

Data Migration Using Logstash

3 min read · Sep 19

1

See more recommendations