Explicación archivos de configuración

Input

```
input {
  file {
    path =>
"/home/g2/g2r12025scrapper/storage/datasets/default/articles.jsonl"
    start_position => "beginning"
    sincedb_path => "/home/g2/ELK/logstash-sincedb"
    mode => "tail"
    codec => json
  }
}
```

`path`: ruta al archivo que se quiere procesar. En este caso, `articles.jsonl` (JSON Lines).

`start_position => "beginning"`: si es la primera vez que se lee este archivo, empieza desde el **principio** del archivo y no desde el final.

`sincedb_path`: ruta al archivo donde Logstash guarda **el estado de lectura**, para no procesar los mismos datos más de una vez.

`mode => "tail"`: Logstash seguirá leyendo nuevas líneas que se agreguen al archivo.

`codec => json`: indica que cada línea del archivo está en formato JSON y debe ser decodificada como tal.

Filter

```
filter {
  ruby {
    code => '
      require "net/http"
      require "uri"
      require "json"

      begin
        uri = URI.parse("http://192.199.1.61:8000/api/embed")
        http = Net::HTTP.new(uri.host, uri.port)
```

```
        req = Net::HTTP::Post.new(uri.path, { "Content-Type" =>
"application/json" })
        req.body = { text: event.get("body") }.to_json
        res = http.request(req)
        if res.code.to_i == 200
          embedding = JSON.parse(res.body)["embedding"]
          event.set("embedding", embedding)
        else
          event.tag("embedding_failed")
        end
      rescue => e
        event.tag("embedding_error")
        event.set("embedding_error_msg", e.message)
      end
    '
  }

  # Extraer la parte de la URL
  grok {
    match => { "url" => "https://%{DATA:fuente}/" }
  }

  # Eliminamos campos innecesarios y ponermos fuente en minuscula
  mutate {
    remove_field => ["host", "log", "event", "@version", "message",
"path", "tags"]
    lowercase => ["fuente"]
  }

  # Convertimos date a timestamp
  date {
    match => ["date", "ISO8601"]
    target => "date"
  }
}
```

Este filtro hace cuatro cosas:

Ruby: Este bloque **ejecuta código Ruby dentro del filtro de Logstash** para enviar el campo `body` de cada evento al endpoint de la API donde está el modelo de embeddings. La API devuelve un vector (`embedding`) que Logstash añade al evento; si falla, marca el evento con tags de error.

Grok: toma la URL y extrae solo el dominio, guardándolo en el campo `fuente`.

Mutate: elimina campos que no necesitas y convierte el valor de `fuente` a minúsculas.

Date: transforma el campo `date` de texto a un timestamp que Elasticsearch puede usar.

Output

```
output {
  elasticsearch {
    hosts =>
["https://192.199.1.59:9200","https://192.199.1.60:9200","https://192.1
99.1.61:9200"]
    api_key => "jAbx3ZoB8_buqiHcbWPg:3ynGrlurs3CySNbLdXXDRg"
    index => "noticias-%{+YYYY.MM.dd}"
    ssl_enabled => true
    ssl_certificate_authorities =>
"/home/g2/ELK/elasticsearch-9.2.1/config/certs/http_ca.crt"
  }

  stdout {
    codec => rubydebug
  }
}
```

Este bloque `output` de Logstash indica a dónde enviar los datos procesados:

1. **Elasticsearch**:
    - Se conecta a los tres hosts que pones.
    - Usa `api_key` para autenticarse.
    - Guarda los documentos en un índice diario llamado `noticias-AAAA.MM.DD`.
    - Usa SSL y el certificado indicado para la conexión segura.
2. **stdout**: imprime los datos en la consola en formato legible (`rubydebug`) para depuración.