**GIỚI THIỆT VỀ EFK STACK (ELASTICSEARCH + FLUENTD + KIBANA) VÀ ỨNG DỤNG VÀO MICROSERVICE APPLICATION**

# **1. Giới thiệu về EFK stack**

Khi xem xét các hệ thống nhỏ và đơn lẻ, việc quản lý log cổ điển được diễn ra một cách thủ công: những người quản lý server nói chung và quản lý log nói riêng sẽ thực hiện việc delpoy sản phẩm lên server, khi cần thì mở file log của server ra xem (tail log), hoặc phải tail nhiều log một lúc để theo dõi (log server, log web, log database, …). Việc làm này thỏa mãn được với các ứng dụng nhỏ và ít người dùng, nhưng lại có một số bất cập như không theo dõi lại được lịch sử, tốn thời gian, công sức, khó phát hiện lỗi hơn.

Với những hệ thống lớn, việc quản lý và phân loại log bằng việc xem file log của server để xác định thông tin của log, phân loại log là khá khó khăn. Cần thiết phải có một công cụ quản lý log một cách tốt hơn, sớm phát hiện những lỗi phát sinh của server hoặc kiểm tra các thông tin về log. Hiện nay cũng có khá nhiều công cụ để quản lý log khác nhau. Qua tìm hiểu thì bộ công cụ EFK có nhiều ưu điểm như phần mềm mã nguồn mở hoàn toàn miễn phí, cung cấp dịch vụ quản lý log rất tốt và dễ sử dụng, vì vậy team quyết định áp dụng EFK vào các dự án của công ty.

***EFK stack*** là cụm từ viết tắt của 3 công cụ kết hợp để thực hiện nhiệm vụ quản lý log: ***Elasticsearch*** (E), ***Fluentd*** (F) và ***Kibana*** (K). Cụ thể:

- ***Elasticsearch (E)***: Công cụ lưu trữ, tìm kiếm, phân tích và truy vấn log.

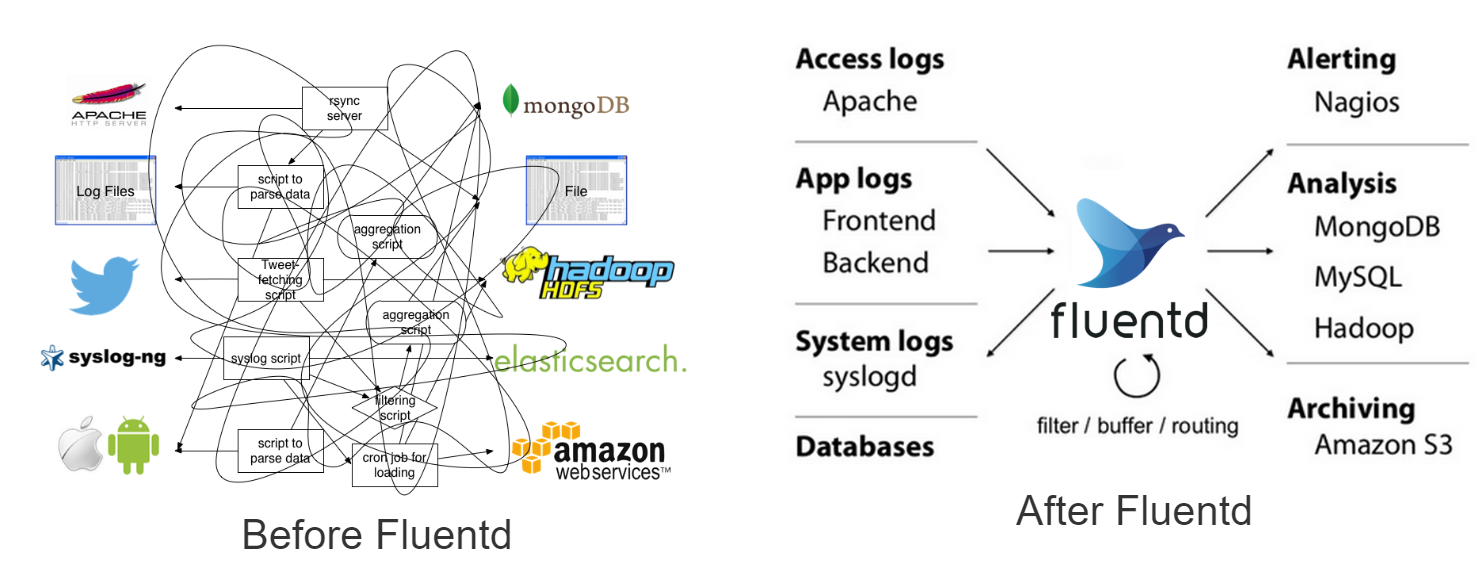
- ***Fluent (F)***: Công cụ “hứng” log (collect log) từ một hoặc nhiều nguồn khác nhau, sau đó có thể forward sang hệ thống khác, hoặc lưu trữ, hoặc ghi ra file.

- ***Kibana (K)***: Đây là công cụ hiển thị log thống kê được từ elasticsearch trên nền web. Thực hiện quản lý và thống kê bằng giao diện trực quan, gần gũi với người sử dụng.



## ***1.1 Fluentd***

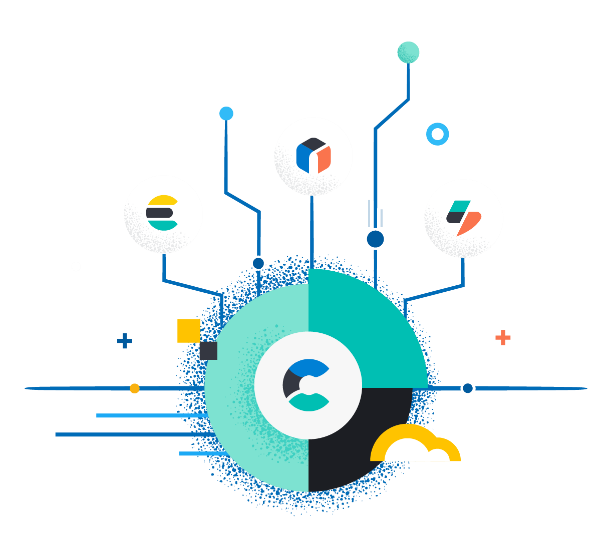
Như đã đề cập phía trên, Fluentd là một công cụ thu thập dữ liệu bằng mã nguồn mở, giúp ta có thể đồng nhất việc gom dữ liệu và sử dụng chúng một cách tốt hơn.



Sau khi ta cấu hình cho một hệ thống nào đó (một webservice, database, log file của hệ thống nào đó, …) sẽ đẩy log thu được cho Fluentd, Fluentd sẽ gom log nhận được từ các hệ thống đó đảm bảo thời gian thực. Với mỗi log thu được Fluentd sẽ gắn cho nó một tag, tag này ta có thể cấu hình được. Việc gắn tag này làm cho việc tiền xử lý và phân loại log được rõ ràng và dễ hơn. Sau bước này, sẽ có nhiều lựa chọn tùy thuộc vào mục đích sử dụng, ta có thể ghi log này vào file, hoặc ghi xuống một cơ sở dữ liệu nào đó để theo dõi, hoặc là forward sang một hệ thống khác để thực hiện xử lý tiếp log mà ta thu được, cụ thể trong trường hợp mà ta đang xét đến đó chính là Elasticsearch. Sau khi thu được log, Fluentd sẽ đẩy log đã được gắn tag sang cho Elasticsearch để tiếp tục xử lý.

## ***1.2 Elasticsearch***

Elasticsearch là một công cụ tìm kiếm (search enginee) dựa trên Apache Lucene với trên 15 năm kinh nghiệm. Nó được xây dựng bằng Java để hoạt động như một server cloud theo cơ chế của RESTful. Điều này giúp nó có thể tương tác và sử dụng bởi rất nhiều ngôn ngữ, cũng chính do đó cũng là điểm yếu của nó khi độ bảo mật không cao.



Elasticsearch sở hữu rất nhiều ưu điểm như tìm kiếm nhanh chóng đảm bảo thời gian thực – realtime searching (hoặc gần như thời gian thực – nearly realtime searching) bằng cách đánh index cho dữ liệu; có khả năng phân tích dữ liệu; lưu trữ NoSQL (sử dụng JSON); sắp xếp kết quả truy vấn theo sự liên quan ([relevance](https://www.elastic.co/guide/en/elasticsearch/guide/current/relevance-intro.html)), …

Vì các ưu điểm nêu trên nên Elasticsearch được sử dụng trong xử lý dữ liệu lớn (big data) và được các “ông lớn” trong làng công nghệ sử dụng.



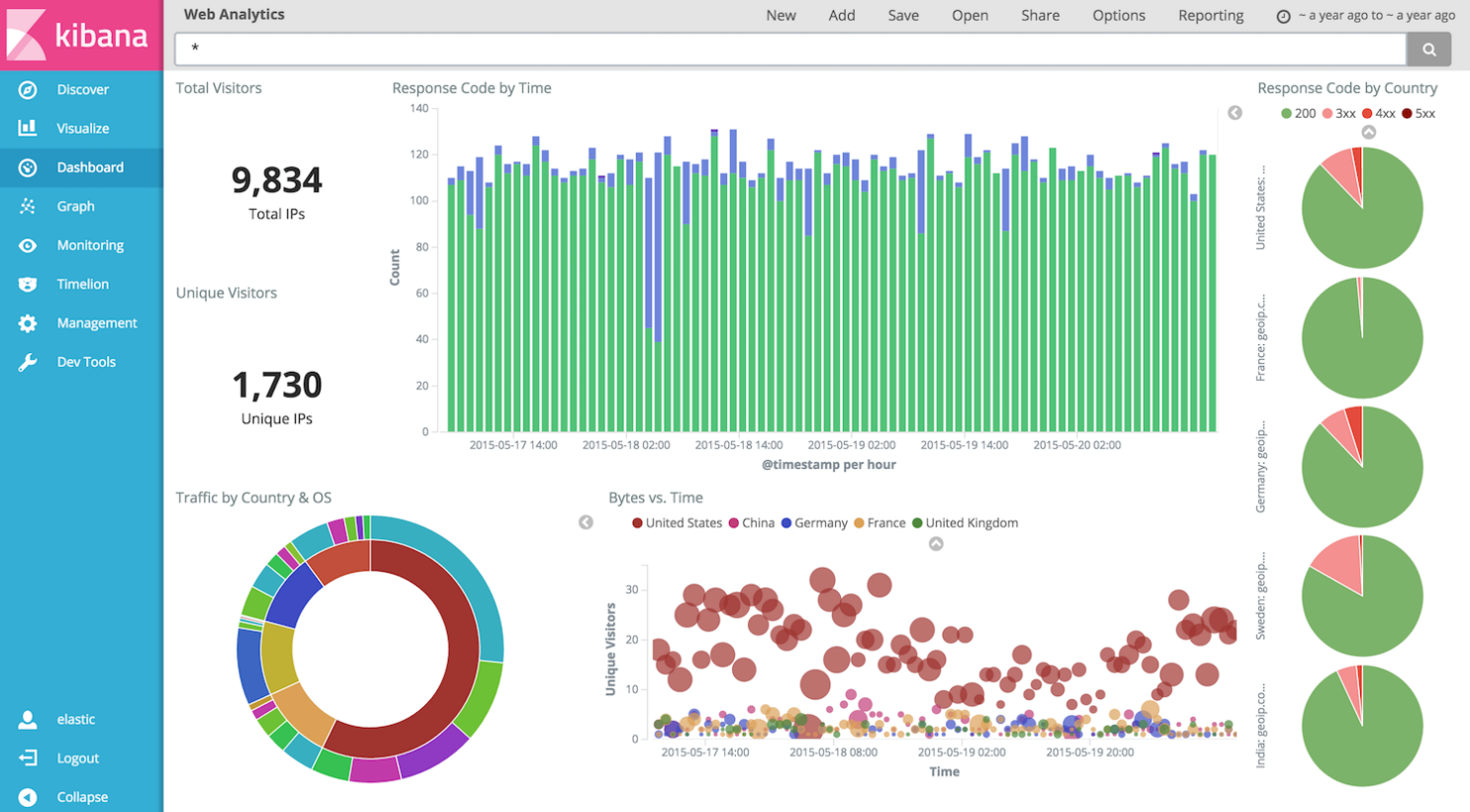
Để giải thích sơ bộ về cơ chế tìm kiếm thời gian thực của Elasticsearch chúng ta cần nói về index. Thông thường khi tìm kiếm một từ nào đó trong một cuốn sách, tương tự với việc tìm kiếm phần tử nào đó trong một bảng, một danh sách, ta thường dùng cách duyệt từ đầu tới cuối, lật từng trang một của cuốn sách và tìm. Nhưng với index thì không thế, việc đánh index cho dữ liệu cũng giống như việc một cuốn sách có phụ lục (index table) lưu các từ xuất hiện trong cuốn sách và trang tương ứng mà nó xuất hiện. Vì vậy nên khi muốn tìm một từ nào đó trong một cuốn sách, ta có thể đi tới phần phụ lục, tìm từ ta muốn và nhảy tới trang được ghi ở phụ lục ta có thể thấy từ đó, tránh phải đọc toàn bộ cuốn sách.

Ngoài cơ chế lưu dữ liệu NoSQL và sử dụng index là tiêu biểu của Elasticsearch còn có rất nhiều thuật toán và cách thức hoạt động phức tạp khác, trong phạm vi seminar sẽ không đề cập, bạn đọc có nhu cầu tìm hiểu xin vui lòng google thêm.

## ***1.3 Kibana***

Là một nền tảng phân tích và trực quan hóa dữ liệu được thiết kế và hoạt động chặt chẽ với Elasticsearch, Kibana cung cấp các tính năng mạnh mẽ và dễ sử dụng như các loại biểu đồ thống kê (biểu đồ đường, biểu đồ quạt, biểu đồ miền, …. ) và hỗ trợ không gian địa lý thích hợp.

Trong phạm vi seminar, Kibana được sử dụng để thống kê các log sau khi phân tích từ Elasticsearch.



## ***1.4 Lợi ích của EFK***

- Quản lý log tập trung, đọc log từ nhiều nguồn.

- Dễ tích hợp với các hệ thống ngoài: Dù server có là ngnix hay apache, hay cơ sở dữ liệu là mySQL hay Oracle, … thì ta đều có thể tích hợp được một cách dễ dàng.

- Miễn phí: Bộ công cụ này đều là open-source và free.

- Khả năng mở rộng và phát triển tốt: Fluentd và Elasticsearch chạy trên nhiều node nên hệ thống EFK cực kì dễ scale. Khi có thêm service, thêm người dùng, muốn log nhiều hơn, bạn chỉ việc thêm node cho Fluentd và Elasticsearch là xong.

- Có bộ lọc và công cụ tìm kiếm mạnh mẽ: Elasticsearch cho phép lưu trữ thông tin kiểu NoSQL, hỗ trợ luôn Full-Text Search nên việc query rất dễ dàng và mạnh mẽ.

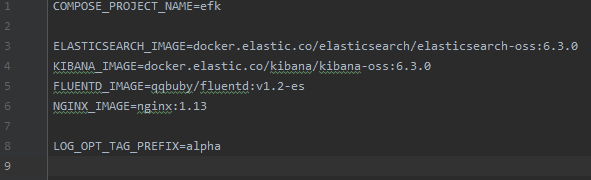
- Cộng đồng hỗ trợ lớn, nhiều tutorial.

# **2. Ứng dụng EFK vào microservice application**

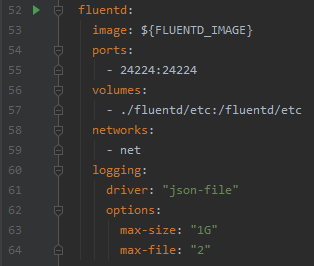
## ***2.1 Cài đặt EFK***

Trong mục này, ta sẽ đi vào phần cài đặt EFK sử dụng docker compose. Trước hết ta cần có file ***docker-compose.yml*** để cấu hình các images cần thiết. Các images này bao gồm: fluentd, elasticsearch và kibana. Link gitlab của project config: <https://git.myitsol.com/truongbb96/efk_config>

Trước tiên, các biến môi trường được định nghĩa ở file ***.env***

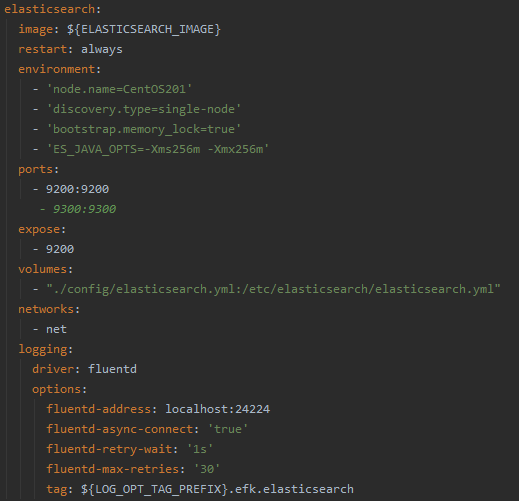


Ta đi vào cấu hình cho Fluent trong file ***docker-compose.yml***



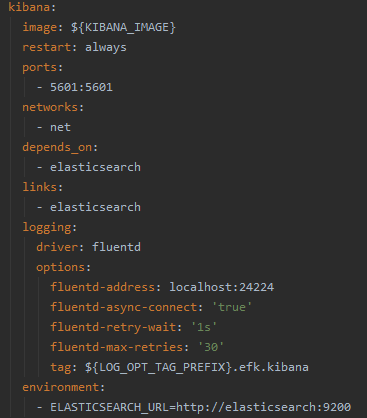
Fluentd sẽ được expose tại cổng 24224 với images được định ngĩa trong file ***.env*** bên trên, với volume cấu hình được định nghĩa ở thư mục ***fluentd/etc*** của project (sẽ đề cập chi tiết phần sau).

Cấu hình cho Elasticsearch:



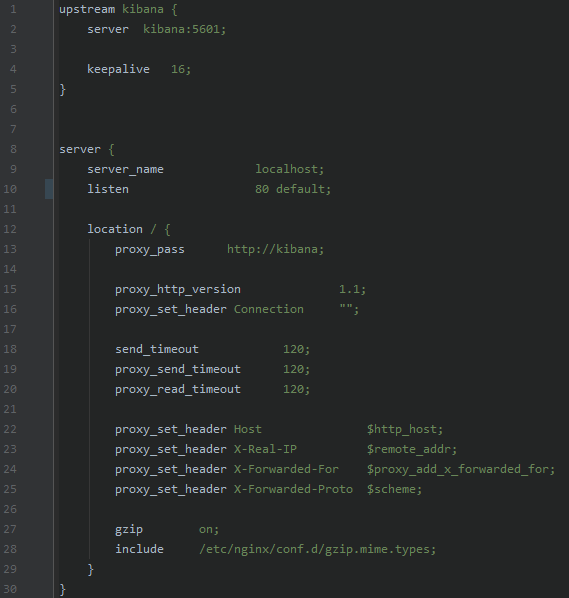
Elasticsearch được expose tại cổng 9200 với images được định nghĩa ở file ***.env*** bên trên. Volume config được lấy trực tiếp từ server ở đường dẫn ***/etc/elasticsearch/elasticsearch.yml***. Điểm chú ý ở đây đó chính là phần config logging, nó sẽ có driver là fluentd gọi tới fluentd được config bên trên với tag là ***alpha.efk.elasticsearch*** (alpha là biến môi trường trong ***.env***).

Cuối cùng là config cho Kibana:

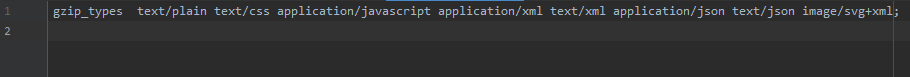


Được public lên cổng 5601, Kibana sẽ phụ thuộc vào elasticsearch (ở mục depends\_on và enviroment) và cũng được cấu hình logging tới fluentd với tag được phân loại là alpha.efk.kibana.

Ngoài ra, chúng ta cần phải cấu hình cho kibana ở file kibana.conf như sau:

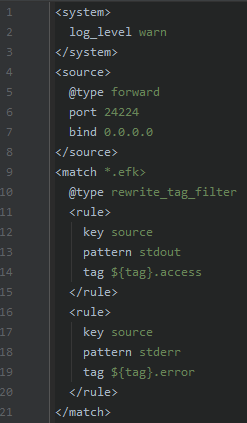


Chú ý rằng mục include sẽ gọi tới file gzip.mine.types, tại đó sẽ định nghĩa ra các loại dữ liệu

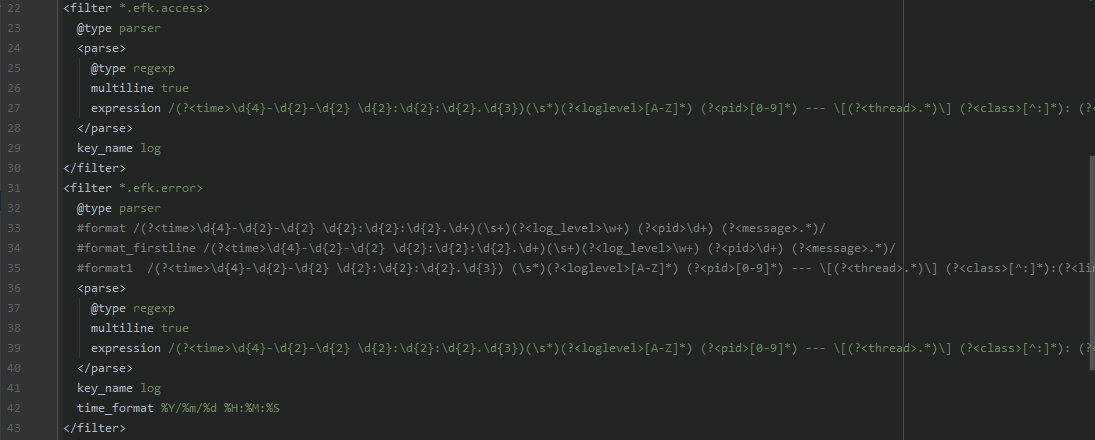


Cuối cùng là file cấu hình fluentd – fluentd.conf, file này sẽ định nghĩa ra các cấu hình cho log, tag, …

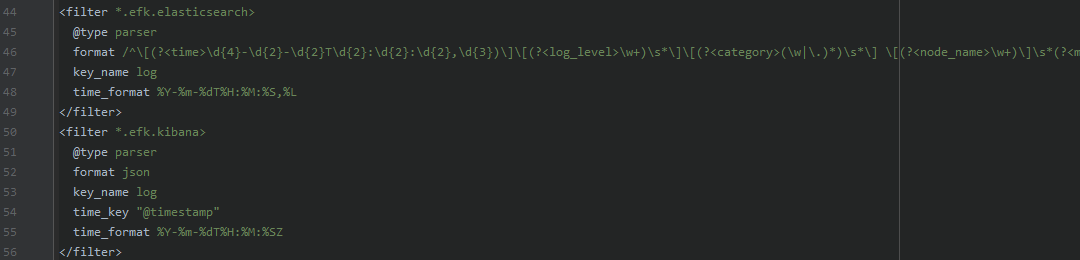
Phần đầu file sẽ là cấu hình chung



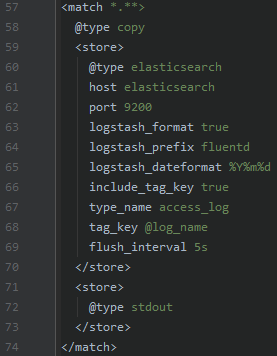
Kiểu của fluentd ở đây là forward (dòng 5), do nó sẽ chuyển log cho elasticsearch để phân tích. Nội dung trong thẻ match sẽ chấp nhận các log có tag là \*.efk và chuyển thành 2 tag khác là access và error tương ứng với stdout và stderr. 2 tag được chuyển này sẽ được cấu hình pattern để bắt log như hình dưới đây:



Ngoài ra, chúng ta còn cấu hình format log cho cả những log nhận được từ elasticsearch system và chính kibana system

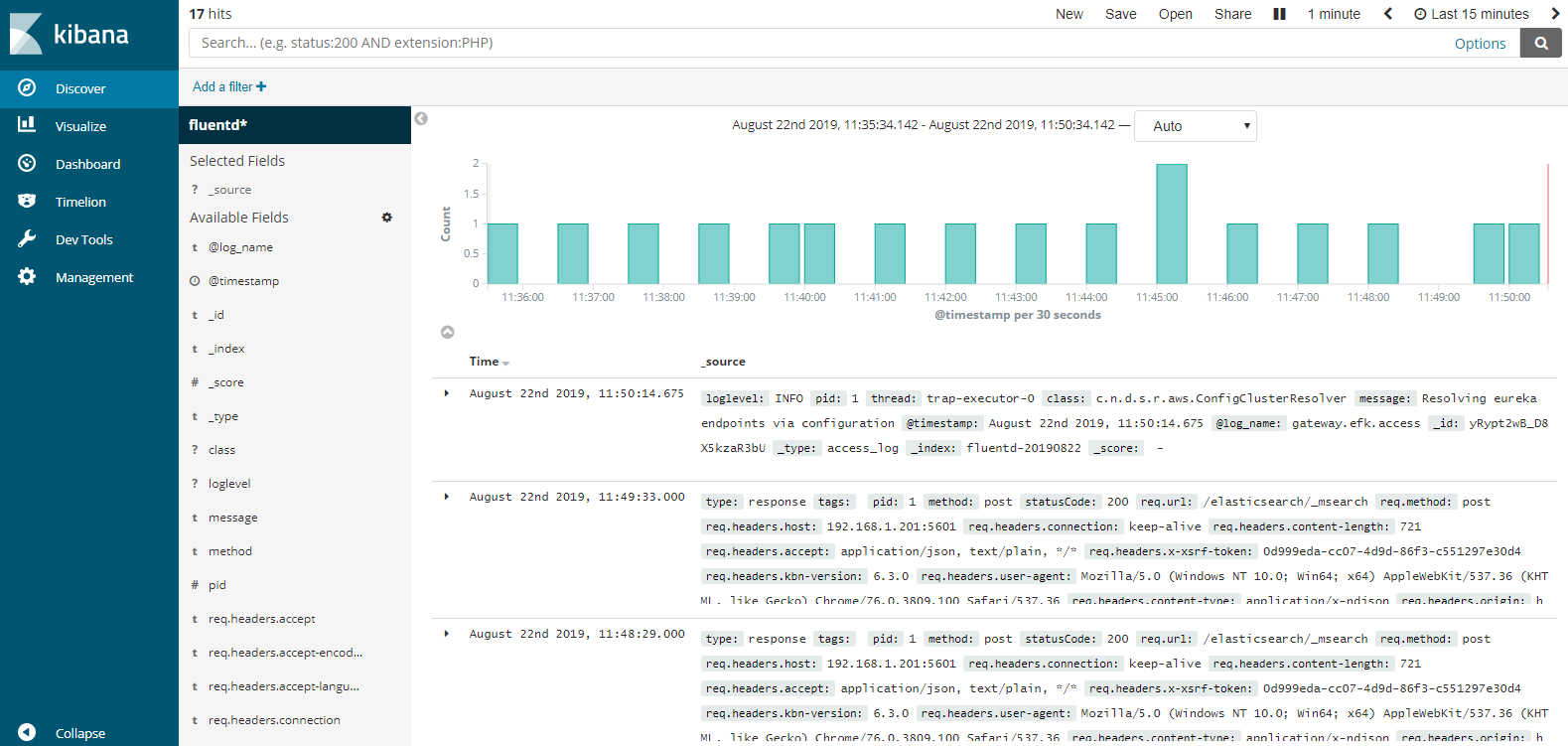


Cuối cùng là mục match

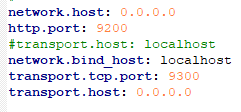


Mục này sẽ có kiểu là copy, và ý nghĩa của nó là copy các log thu được để lưu vào elasticsearch ở cổng 9200 với định dạng của logstash (logstash\_format), tiền tố là fluentd (logstash\_prefix), truyền cả tag key được gán (include\_tag\_key) và đẩy đi cứ 5s một lần.

Sau khi cấu hình project EFK xong, ta thực hiện lệnh docker-compose up để EFK hoạt động. Vào thử đường dẫn 192.168.1.201:5601 (cổng của kibana) để xem kết quả thu được:

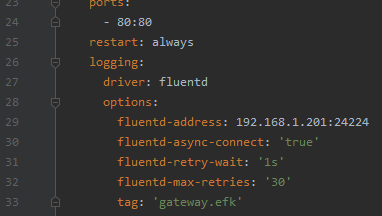


Một điều lưu ý khác khi thực hiện xong tất cả các bước trên, ta cần thêm config cho elasticsearch ở file ***elasticsearch.yml*** (tại đường dẫn etc/elasticsearch/elasticsearch.yml) như sau:

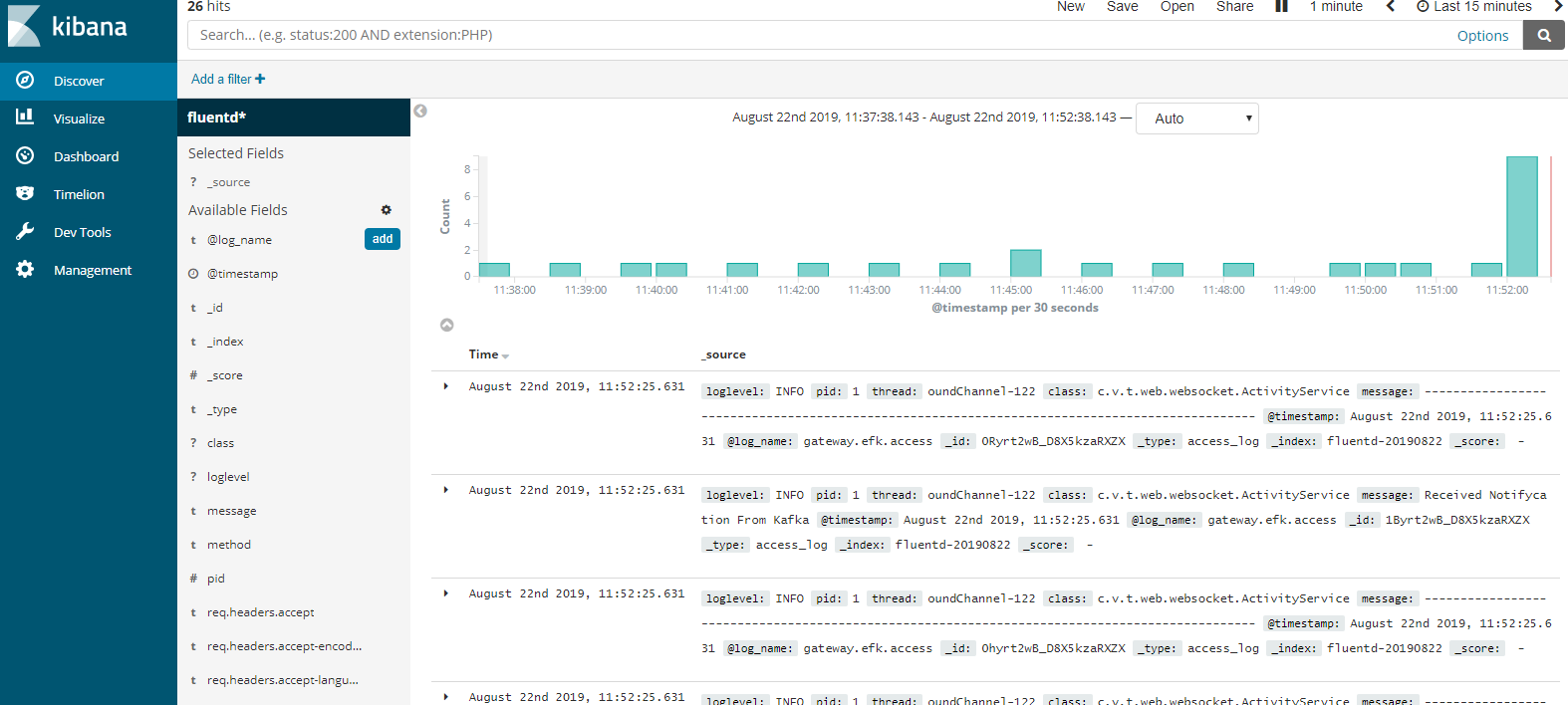


## ***2.2 Cấu hình một service gửi log tới EFK***

Trong mục này, team sẽ thực hiện demo với prioject gateway đã được áp dụng CI lần trước. Tất cả công việc cần làm ở đây là thêm đoạn cấu hình logging vào file app.yml (đóng vai trò là file docker-compose của project) để log của project này sẽ được hứng bởi fluentd:



Để xem kết quả, ta truy cập vào Kibana ở địa chỉ 192.168.1.201:5601



# **3. Reference**

[1] <https://blog.vietnamlab.vn/2018/05/30/quan-ly-log-voi-logstash-elasticsearch-kibana/>

[2] <https://toidicodedao.com/2018/03/20/elk-stack-logging/>

[3] <https://www.linkedin.com/pulse/big-data-t%E1%BB%95ng-quan-v%E1%BB%81-elasticsearch-donald-trung-manh-nguyen>

[4] <https://viblo.asia/p/elasticsearch-khai-niem-va-cac-cau-truy-van-co-ban-djeZ1VwRlWz>

[5] <https://hicubex2k.wordpress.com/2016/11/03/tim-hieu-ve-fluentd/>

[6] <https://stolennguyen.website/kibana-la-gi/>

[7]