

**ĐẠI HỌC QUỐC GIA HÀ NỘI  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

-----

**TRẦN VĂN LINH**

**NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG PHÁT HIỆN SỰ CỐ  
HỆ THỐNG DỰA TRÊN CÔNG NGHỆ ELK (ELASTICSEARCH,  
LOGSTASH VÀ KIBANA)**

**LUẬN VĂN THẠC SĨ  
NGÀNH HỆ THỐNG THÔNG TIN**

**Hà Nội - 2019**

**ĐẠI HỌC QUỐC GIA HÀ NỘI**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

**TRẦN VĂN LINH**

**NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG PHÁT HIỆN SỰ CỐ**  
**HỆ THỐNG DỰA TRÊN CÔNG NGHỆ ELK**  
**(ELASTICSEARCH, LOGSTASH VÀ KIBANA)**

Ngành: Hệ thống thông tin

Chuyên ngành: Hệ thống thông tin

Mã số: 8480104.01

**LUẬN VĂN THẠC SĨ**  
**NGÀNH HỆ THỐNG THÔNG TIN**

**GIẢNG VIÊN HƯỚNG DẪN:**

**PGS.TS. Nguyễn Hà Nam**

**Hà Nội - 2019**

## LỜI CẢM ƠN

Để có thể hoàn thiện được luận văn thạc sĩ của mình, trước tiên, tôi xin bày tỏ lòng biết ơn sâu nhất tới thầy – PGS.TS. Nguyễn Hà Nam (bộ môn Các hệ thống thông tin – trường Đại học Công nghệ – Đại học Quốc gia Hà Nội). Sự gần gũi và nhiệt tình hướng dẫn của thầy là nguồn động lực rất lớn đối với tôi trong suốt thời gian thực hiện luận văn.

Tôi cũng xin được gửi lời cảm ơn chân thành nhất tới tất cả các thầy, cô trong bộ môn Các hệ thống thông tin, cũng như các thầy, cô trong khoa Công nghệ thông tin – trường Đại học Công nghệ – Đại học Quốc gia Hà Nội đã nhiệt tình giảng dạy, cung cấp cho chúng tôi những kiến thức, kinh nghiệm không chỉ trong học tập mà còn trong cuộc sống hàng ngày.

Đồng thời tôi cũng xin được gửi lời cảm ơn đến cha mẹ, người thân trong gia đình, các bạn học viên, đồng nghiệp đã giúp đỡ, động viên, tạo điều kiện tốt nhất cho tôi trong suốt khóa học tại Trường Đại học Công nghệ – Đại học Quốc gia Hà Nội để tôi có thể hoàn thiện tốt luận văn thạc sĩ của mình.

Hà Nội, tháng    năm  
Học viên

Trần Văn Linh

**LỜI CAM ĐOAN**

Tôi xin cam đoan luận văn tốt nghiệp “*Nghiên cứu giải pháp tự động phát hiện sự cố hệ thống dựa trên công nghệ ELK (ElasticSearch, Logstash và Kibana)*” là công trình nghiên cứu thực sự của bản thân, được thực hiện trên cơ sở nghiên cứu lý thuyết, kiến thức chuyên ngành, nghiên cứu khảo sát tình hình thực tiễn và dưới sự hướng dẫn khoa học của PGS.TS. Nguyễn Hà Nam. Các kết quả được viết chung với các tác giả khác đều được sự đồng ý của tác giả trước khi đưa vào luận văn. Những phần tham chiếu, trích dẫn trong luận văn đều được nêu rõ trong phần tài liệu tham khảo. Các số liệu, kết quả trình bày trong luận văn là hoàn toàn trung thực. Nếu sai tôi xin chịu hoàn toàn trách nhiệm và chịu mọi kỷ luật của khoa và nhà trường đề ra.

Hà Nội, tháng      năm 201

Học viên

Trần Văn Linh

## MỤC LỤC

<b>LỜI CẢM ƠN</b> .....	2
<b>LỜI CAM ĐOAN</b> .....	3
<b>MỤC LỤC</b> .....	4
<b>DANH MỤC CÁC TỪ VIẾT TẮT</b> .....	6
<b>DANH MỤC HÌNH VẼ</b> .....	7
<b>MỞ ĐẦU</b> .....	8
<b>CHƯƠNG I: GIỚI THIỆU BÀI TOÁN VÀ LỰA CHỌN CÔNG NGHỆ</b> .....	12
1.1. Một số khái niệm.....	12
1.2. Giới thiệu bài toán.....	12
1.3. Lựa chọn công nghệ.....	13
1.4. Tìm hiểu nền tảng công nghệ ELK.....	21
1.4.1. Giới thiệu ELK.....	21
1.4.2. Elasticsearch.....	21
1.4.3. Logstash.....	33
1.4.4. Kibana.....	41
1.5. Kết luận.....	41
<b>CHƯƠNG II: PHÂN TÍCH, THIẾT KẾ, XÂY DỰNG HỆ THỐNG QUẢN LÝ LOG TẬP TRUNG CHO TẬP ĐOÀN BẢO VIỆT</b> .....	42
2.1. Hiện trạng hạ tầng CNTT Bảo Việt.....	42
2.1.1. Hiện trạng dịch vụ.....	42
2.1.2. Hiện trạng hạ tầng máy chủ.....	43
2.1.3. Hiện trạng nền tảng hệ điều hành và phần mềm.....	43
2.1.4. Hiện trạng mô hình hạ tầng hệ thống CNTT.....	44
2.1.5. Hiện trạng quản lý, giám sát hệ thống.....	45
2.2. Kiến trúc giải pháp.....	45
2.2.1. Mô hình tổng thể giải pháp.....	46
2.2.2. Mô hình luồng dữ liệu.....	49
2.2.3. Mô hình trao đổi dữ liệu với các hệ thống khác.....	50
2.3. Kết luận.....	50
<b>CHƯƠNG III: XÂY DỰNG THỬ NGHIỆM HỆ THỐNG QUẢN LÝ LOG TẠI BẢO VIỆT</b> .....	51
3.1. Môi trường thử nghiệm.....	51
3.2. Mô hình và cấu hình thử nghiệm.....	53
3.3. Kết quả đạt được.....	59
3.4. Đánh giá kết quả.....	62
3.5. Các vấn đề còn tồn tại và hướng phát triển.....	63

3.5.1.	Vấn đề sử dụng nhiều tài nguyên máy chủ.....	63
3.5.2.	Vấn đề thất lạc dữ liệu log.....	63
3.5.3.	Hướng phát triển giải quyết vấn đề .....	63
3.6.	Kết luận .....	64
<b>KẾT LUẬN</b>	.....	<b>65</b>
<b>TÀI LIỆU THAM KHẢO</b>	.....	<b>67</b>

**DANH MỤC CÁC TỪ VIẾT TẮT**

CSDL	Cơ sở dữ liệu
ELK	ElasticSearch, Logstash, Kibana
ETL	Extract, Transform, Load
ESB	Enterprise Service Bus
EAI	Enterprise application integration
EDR	Enterprise data replication
VSM	Vector Space Model
CNTT	Công nghệ thông tin
BI	Business Intelligence

## DANH MỤC HÌNH VẼ

Hình 1.1 : Một số nền tảng công nghệ được sử dụng để quản lý log.....	14
Hình 1.2 : Mô hình Massive Parallel Processing .....	23
Hình 1.3 : Mô hình cụm Cluster của ElasticSearch.....	23
Hình 1.4 : So sánh các thành phần của ElasticSearch với Cơ sở dữ liệu quan hệ.....	24
Hình 1.5 : Biểu đồ Term Frequency của mô hình BM25 và TF/IDF .....	29
Hình 1.6 : Tiến trình phân tích từ tố (Analysis) trong ElasticSearch .....	32
Hình 1.7 : Giới thiệu logstash.....	33
Hình 1.8 : Tiến trình ETL.....	35
Hình 1.9 : Các tiến trình hoạt động của Logstash .....	35
Hình 1.10 : Cơ chế hoạt động Pull và Push của Logstash.....	35
Hình 1.11 : Giới thiệu Kibana .....	41
Hình 2.1 : Cơ cấu tổ chức và mô hình dịch vụ tại Tập đoàn Bảo Việt.....	42
Hình 2.2 : Hiện trạng mô hình hạ tầng CNTT tại Bảo Việt .....	44
Hình 2.3 : Mô hình tổng thể giải pháp.....	46
Hình 2.4 : Mô hình hoạt động của Logstash.....	47
Hình 2.5 : Mô hình luồng dữ liệu giải pháp.....	49
Hình 2.6 : Mô hình trao đổi dữ liệu với các hệ thống khác .....	50
Hình 3.1: Mô hình hệ thống dịch vụ BVCare.....	52
Hình 3.2 : Mô hình cấu hình thử nghiệm giải pháp ELK cho dịch vụ BVCare .....	53
Hình 3.3 : Sơ đồ khối luồng xử lý dữ liệu log với Logstash.....	54
Hình 3.4 : Sơ đồ khối bước “Lọc và chuẩn hóa dữ liệu log” thực hiện trong Logstash .....	55
Hình 3.5 : Email cảnh báo hết dung lượng lưu trữ.....	60
Hình 3.6 : Biểu đồ thống kê mã lỗi gặp phải với hệ thống BVCare.....	61
Hình 3.7 : Email cảnh báo địa chỉ lạ kết nối tới cơ sở dữ liệu hệ thống BVCare.....	61
Hình 3.8 : Biểu đồ thống kê địa chỉ và người dùng kết nối tới cơ sở dữ liệu BVCare.....	62
Hình 3.9 : Mô hình hướng phát triển của giải pháp .....	64



## MỞ ĐẦU

Trong thời đại công nghệ số, các dịch vụ mua bán, trao đổi truyền thống dần được thay thế bởi thương mại điện tử. Đặc biệt với các giao dịch tài chính như chuyển tiền ngân hàng, mua bán các hợp đồng bảo hiểm, chứng khoán đều được thực hiện trên internet thông qua các dịch vụ công nghệ thông tin mà các tổ chức tài chính, ngân hàng, bảo hiểm xây dựng để cung cấp cho khách hàng.

Chất lượng dịch vụ công nghệ thông tin của một tổ chức, doanh nghiệp mang ý nghĩa sống còn và là lợi thế cạnh tranh không hề nhỏ cho doanh nghiệp trên thương trường; đặc biệt đối với các doanh nghiệp hoạt động trong lĩnh vực tài chính, bảo hiểm, ngân hàng, khi mà các hệ thống phải hoạt động liên tục 24/7 để đáp ứng nhu cầu sử dụng dịch vụ của khách hàng. Khi chất lượng dịch vụ công nghệ thông tin được nâng cao, trải nghiệm của người dùng tốt thì sẽ thu hút và giữ được khách hàng và ngược lại.

Trong quá trình hoạt động, hệ thống công nghệ thông tin của doanh nghiệp có thể gặp sự cố bất cứ lúc nào. Các tổ chức phải đối mặt với sự cố ngừng hoạt động dịch vụ và các mối đe dọa bảo mật khác nhau, việc giám sát toàn bộ nền tảng ứng dụng là điều cần thiết để hiểu rõ nguồn gốc của mối đe dọa hoặc nguyên nhân xảy ra sự cố, cũng như để xác minh các sự kiện, log và dấu vết để hiểu hành vi của hệ thống tại thời điểm đó và dự đoán cũng như có hành động khắc phục kịp thời. Giám sát và phân tích dữ liệu log là rất quan trọng đối với bất kỳ tổ chức hoạt động CNTT nào để xác định các nỗ lực xâm nhập trái phép, theo dõi hiệu suất ứng dụng, cải thiện sự hài lòng của khách hàng, tăng cường bảo mật chống lại các cuộc tấn công mạng, thực hiện phân tích nguyên nhân gốc và phân tích hành vi, hiệu suất, đo lường và số liệu dựa trên phân tích dữ liệu log. Việc phát hiện và xử lý sớm các lỗi dẫn tới nguy cơ xảy ra sự cố hệ thống có ý nghĩa rất quan trọng đối với chất lượng dịch vụ mà doanh nghiệp cung cấp ra. Nhiều doanh nghiệp đang thực hiện công việc này một cách thủ công – một nhóm cán bộ phụ trách trực hạ tầng sẽ thực hiện đọc log của từng hệ thống riêng lẻ và tìm kiếm theo các từ khóa về lỗi có thể xảy ra, khi từ khóa được tìm thấy, cán bộ trực hạ tầng thực hiện gửi thư điện tử (email) thông báo tới cán bộ quản trị dịch vụ để khắc phục lỗi. Việc tìm kiếm lỗi và gửi email thông báo theo cách thủ công này dẫn tới việc chậm trễ và thiếu sót trong quá trình phát hiện và xử lý lỗi, ảnh hưởng trực tiếp đến hoạt động kinh doanh của doanh nghiệp. Bài toán đặt ra là cần phải tự động hóa công việc này để phát hiện và thông báo lỗi tới cán bộ quản trị một cách nhanh nhất để giảm thiểu tối đa thời gian ảnh hưởng đến khả năng cung cấp dịch vụ của doanh nghiệp.

Mục tiêu của đề tài là tập trung vào nghiên cứu, xây dựng giải pháp công nghệ để giải quyết bài toán trên cho doanh nghiệp. Giải quyết được bài toán này mang ý nghĩa hết sức to lớn cho doanh nghiệp, nó không những phục vụ cho các hoạt động

kinh doanh của doanh nghiệp mà còn tăng vị thế, uy tín của doanh nghiệp khi chất lượng các dịch vụ mà doanh nghiệp cung cấp được nâng cao.

Có khá nhiều nền tảng công nghệ đáp ứng được cho bài toán quản lý log, có thể kể ra một số nền tảng nổi trội như Splunk, Graylog, Loggly, Solarwind hoặc bản thân nội tại các hệ thống như hệ điều hành Window, Linux, Database,... cũng có phần quản lý logs của riêng mình. Tuy nhiên điểm yếu của các nền tảng công nghệ kể trên hoặc là không hỗ trợ quản lý tập trung, không có nền tảng tìm kiếm đủ mạnh, không hỗ trợ mở rộng, không hỗ trợ phân tích logs hoặc khi sử dụng phải trả một khoản chi phí rất cao (phần mềm thương mại). Đề tài tập trung nghiên cứu giải pháp công nghệ ELK do công nghệ này hài hòa giữa chi phí đầu tư và khả năng đáp ứng các tiêu chí như mã nguồn mở, hỗ trợ mở rộng theo chiều ngang (Clusters), có một nền tảng tìm kiếm mạnh mẽ (ElasticSearch), hỗ trợ phân tích số liệu thu thập được (Analytic), quản lý logs tập trung, tìm kiếm và thông báo lỗi một cách tự động.

### **1. Tính cấp thiết của đề tài**

Tại Việt Nam, các dịch vụ Công nghệ thông tin phục vụ nghiệp vụ kinh doanh đang trở thành xương sống của các doanh nghiệp. Nền kinh tế hội nhập đòi hỏi các doanh nghiệp cần phải cung cấp được các dịch vụ với chất lượng cao nhất có thể để cạnh tranh được với các đối thủ trong và ngoài nước.

Một hệ thống dịch vụ Công nghệ thông tin phục vụ nghiệp vụ kinh doanh đòi hỏi dịch vụ đó phải có tốc độ xử lý nhanh, phục vụ được nhiều người dùng đồng thời và phải luôn sẵn sàng 24/7.

Bất kỳ một hệ thống dịch vụ Công nghệ thông tin nào cũng tiềm ẩn nhiều rủi ro gây gián đoạn hoạt động kinh doanh của Tổ chức, Doanh nghiệp. Để có thể cung cấp dịch vụ Công nghệ thông tin với chất lượng cao nhất, ngoài đội ngũ cán bộ có trình độ chuyên môn tốt để phát triển và vận hành các dịch vụ công nghệ thông tin thì doanh nghiệp cũng cần phải có các giải pháp để giúp giảm thiểu sự cố, rủi ro gây gián đoạn dịch vụ.

Tại Tập đoàn Bảo Việt, Trung tâm Công nghệ thông tin cung cấp hàng trăm dịch vụ Công nghệ thông tin cho các đơn vị thành viên phục vụ nghiệp vụ kinh doanh. Số lượng máy chủ và phần mềm lên tới con số hàng nghìn. Kiểm soát được mọi diễn biến, thay đổi của hệ thống từng phút, từng giờ sẽ giúp đội ngũ kỹ thuật phát hiện sớm những rủi ro tiềm ẩn có thể gây sự cố gián đoạn dịch vụ, ảnh hưởng tới kết quả kinh doanh của các công ty thành viên và toàn Tập đoàn. Do đó việc nghiên cứu và xây dựng một hệ thống quản lý log tập trung cho các hệ thống của Tập đoàn nhằm mục đích phát hiện sớm được các rủi ro tiềm ẩn là vấn đề hết sức cần thiết và cấp bách mà theo tác giả không chỉ cần thiết cho riêng Tập đoàn Bảo Việt mà còn cho bất cứ doanh nghiệp nào sử dụng các dịch vụ Công nghệ thông tin vào phục vụ sản xuất kinh doanh.

## 2. Đối tượng nghiên cứu

Bất kỳ một nền tảng, một giải pháp công nghệ nào áp dụng cho bài toán quản lý logs tập trung đều phải giải quyết được 3 vấn đề chính:

- Thu thập logs
- Phân tích và tìm kiếm
- Giám sát và cảnh báo

Từ đó, chúng tôi xác định đối tượng nghiên cứu của đề tài là một số kỹ thuật, công nghệ tích hợp dữ liệu, truy hồi thông tin để làm nền tảng cơ sở lý thuyết. Một số nền tảng công nghệ sử dụng cho bài toán quản lý logs tập trung, trong đó tập trung vào nghiên cứu **công nghệ mã nguồn mở ELK** (ElasticSearch, LogStash và Kibana) và đưa vào áp dụng cho Tập đoàn Bảo Việt.

## 3. Mục tiêu nghiên cứu

Tìm hiểu được một số kỹ thuật và công nghệ tích hợp dữ liệu, truy hồi thông tin. Một số nền tảng công nghệ nổi bật được sử dụng trong bài toán quản lý log tập trung, trong đó đi sâu vào nghiên cứu công nghệ mã nguồn mở ELK (ElasticSearch, LogStash và Kibana) để đưa vào áp dụng tại Tập đoàn Bảo Việt.

Để thực hiện mục tiêu đã đề ra, chúng tôi phân chia luận văn thành 3 chương được cấu trúc như sau:

**Chương I:** Giới thiệu bài toán và lựa chọn công nghệ.

**Chương II:** Phân tích, thiết kế, xây dựng hệ thống quản lý log tập trung cho tập đoàn Bảo Việt dựa trên nền tảng công nghệ ELK.

**Chương III:** Thử nghiệm giải pháp và đánh giá các kết quả đạt được, các điểm còn hạn chế và hướng phát triển kế tiếp.

## 4. Các công việc cần thực hiện

Để đạt được mục tiêu nghiên cứu đã đề ra, chúng tôi đã hoạch định các công việc cần phải thực hiện theo các bước như sau:

- Tìm hiểu nền tảng lý thuyết về công nghệ tích hợp dữ liệu và hệ truy hồi thông tin.
- Tìm hiểu về một số nền tảng công nghệ thông dụng hiện nay được sử dụng cho bài toán quản lý dữ liệu log tập trung.
- So sánh các giải pháp công nghệ và lựa chọn giải pháp công nghệ phù hợp để triển khai bài toán quản lý dữ liệu log tại Tập đoàn Bảo Việt.
- Tìm hiểu chi tiết nền tảng công nghệ đã lựa chọn về mô hình, cách thức hoạt động, thành phần, cài đặt, cấu hình, cú pháp lập trình, ...

- Hệ thống hóa lại hiện trạng hệ thống phần mềm tại Tập đoàn Bảo Việt để có thể phân tích, thiết kế được mô hình triển khai hệ thống quản lý dữ liệu log tập trung.
- Triển khai thử nghiệm hệ thống quản lý dữ liệu log tại Bảo Việt.
- Đánh giá kết quả đạt được và đường hướng phát triển trong tương lai.

## CHƯƠNG I: GIỚI THIỆU BÀI TOÁN VÀ LỰA CHỌN CÔNG NGHỆ

### 1.1. Một số khái niệm

Một số khái niệm liên quan đến hệ thống phần mềm

#### a. Sự kiện (*Event*)

Một sự kiện là một thay đổi quan sát được đối với hành vi của hệ thống phần mềm. Sự kiện xảy ra có thể là hành vi thông thường hoặc bất thường của hệ thống. Sự kiện bất thường của hệ thống có thể dẫn đến sự cố cho hệ thống.

#### b. Dữ liệu log sự kiện (*Event log*)

Log sự kiện là một tệp dữ liệu lưu trữ lại các sự kiện xảy ra trong hệ điều hành hoặc phần mềm. Dữ liệu log lưu trữ mọi dấu vết hoạt động hiện tại và quá khứ của hệ thống phần mềm, cho biết tình trạng hoạt động, các lỗi xảy ra và vấn đề về an ninh, bảo mật đối với hệ thống.

#### c. Sự cố (*Incident*)

Một sự kiện không nằm trong các hành vi thông thường của hệ thống phần mềm, gây gián đoạn hoạt động của dịch vụ hoặc giảm chất lượng dịch vụ được gọi là sự cố.

Dữ liệu log bản thân nó bao hàm rất rộng như dữ liệu log sự kiện (event log), dữ liệu log giao dịch (transaction log) hay dữ liệu log thông điệp (message log). Sau đây, trong phạm vi luận văn, chúng tôi quy ước rằng khi đề cập đến dữ liệu log đồng nghĩa với dữ liệu log sự kiện của hệ thống phần mềm.

### 1.2. Giới thiệu bài toán

Bài toán quản lý dữ liệu log tập trung, tự động phát hiện lỗi, sự cố không chỉ là bài toán cần giải quyết của riêng Tập đoàn Bảo Việt mà còn của bất kỳ tổ chức, doanh nghiệp cung cấp dịch vụ Công nghệ thông tin nào. Trong thời buổi kinh tế ngày nay, chất lượng dịch vụ mang lại giá trị rất lớn cho doanh nghiệp trên thương trường. Các hệ thống dịch vụ công nghệ thông tin cần phải hoạt động liên tục 24/7 và phải hạn chế tối đa việc gặp sự cố gây gián đoạn dịch vụ.

Đối với Bảo Việt thì yêu cầu trên càng trở nên cấp thiết hơn bao giờ hết, bởi Bảo Việt là một tập đoàn hoạt động trong hầu hết các lĩnh vực tài chính, bảo hiểm, ngân hàng, chứng khoán, đầu tư, ... Số lượng dịch vụ công nghệ thông tin mà Tập đoàn đang quản lý và cung cấp cho các đơn vị thành viên sử dụng phục vụ hoạt động kinh doanh lên tới hàng trăm dịch vụ với hàng nghìn hệ thống máy chủ, máy trạm hoạt động suốt ngày đêm, và với đặc thù của những dịch vụ cần hoạt động 24/7 như dịch vụ Bảo lãnh y tế mà các bệnh viện liên kết với Bảo Việt đang sử dụng không ngừng nghỉ thì đòi hỏi dịch vụ phải ít gặp sự cố nhất có thể, thời gian gián đoạn dịch vụ là ngắn nhất có thể.

Qua khảo sát hiện trạng tại Tập đoàn Bảo Việt, công việc quản lý và giám sát log của các hệ thống dịch vụ Công nghệ thông tin còn thủ công, phân tán và thiếu tính chính xác, dẫn tới nguy cơ xảy ra sự cố luôn tiềm ẩn, ảnh hưởng đến hoạt động kinh doanh của các đơn vị.

Hai vấn đề lớn cần giải quyết tại Tập đoàn Bảo Việt cho bài toán quản lý log là:

- Cần quản lý log tập trung phục vụ bài toán tìm kiếm, phân tích hiệu quả nhất và nhanh nhất.
- Cần tự động phát hiện các lỗi trong dữ liệu log để kịp thời khắc phục trước khi sự cố xảy ra với hệ thống, giảm thiểu các công việc thủ công thiếu chính xác trong việc giám sát và phân tích dữ liệu log.

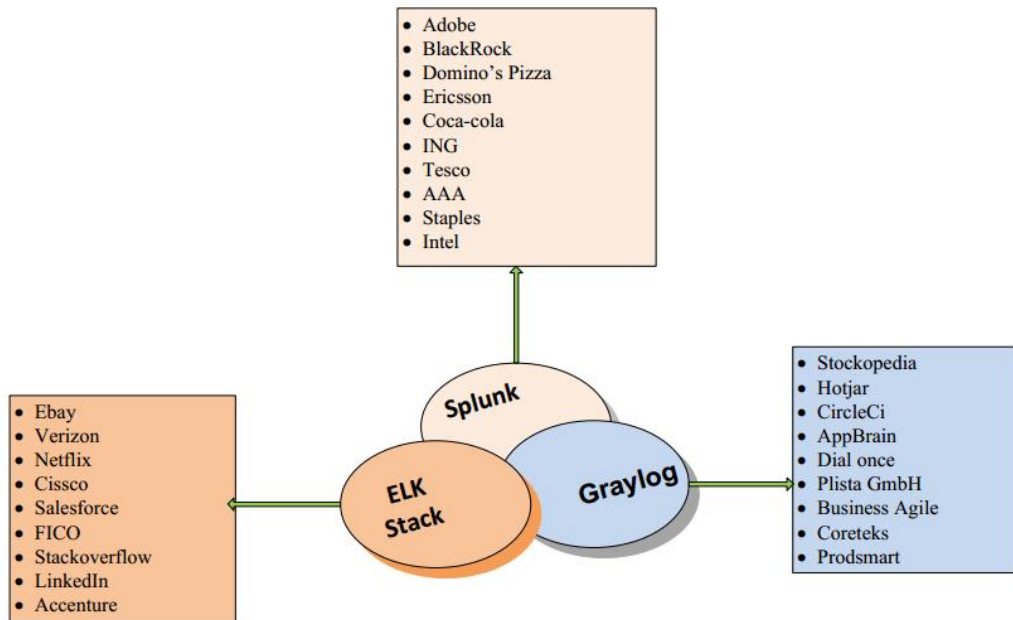
Thách thức khi thực hiện bài toán quản lý log tập trung tại Tập đoàn Bảo Việt:

- Số lượng máy chủ là rất lớn và đa dạng nền tảng, có những thiết bị rất đặc thù: Window Server, Linux, HP-UX, Solaris, VMWare, ...
- Số lượng phần mềm cũng rất lớn và đa dạng: Database, Phần mềm lớp giữa, Web Server, Application Server, Http Server, ...
- Việc tích hợp dữ liệu log phải ít ảnh hưởng nhất đến hoạt động và hiệu năng của dịch vụ.

### **1.3. Lựa chọn công nghệ**

Bài toán quản lý dữ liệu log tập trung và cảnh báo lỗi tự động có thể được thực hiện bằng nhiều giải pháp công nghệ khác nhau bao gồm cả các giải pháp phần mềm thương mại và phần mềm mã nguồn mở, điển hình trong số đó phải kể đến như Splunk, Graylog và ELK. Các tính năng nổi trội của các giải pháp giám sát và phân tích dữ liệu log hệ thống ngày nay phải kể đến như: Khả năng tìm kiếm mạnh mẽ, xây dựng được màn hình giám sát thời gian thực, báo cáo, cảnh báo ngưỡng, phân tích dữ liệu lịch sử, truy tìm vết, ...

Có thể thống kê một số tổ chức nổi bật đang sử dụng các giải pháp trên vào quản lý và phân tích dữ liệu log:



Hình 1.1 : Một số nền tảng công nghệ được sử dụng để quản lý log

Mỗi một giải pháp đều có những ưu, nhược điểm và mức độ phù hợp riêng. Dưới đây là bảng so sánh về một số tính năng của ba giải pháp trên:

Bảng 1.1 : Bảng so sánh các tính năng của 3 pháp quản lý dữ liệu log thông dụng trên thị trường.

Tính năng	ELK Stack	Splunk	Graylog
Bản quyền	- Mã nguồn mở - Thương mại	Thương mại	- Mã nguồn mở - Thương mại
Ngôn ngữ	Java, JRuby, NodeJS	C++, Python	Java
Định dạng dữ liệu	JSON	JSON, CSV, Text	GLEF
Độ phức tạp cài đặt	Cần cài đặt 3 thành phần: - Elasticsearch - Logstash - Kibana Độ phức tạp trung bình.	Chỉ cần cài đặt 1 thành phần Splunk Server, dễ dàng hơn để cài đặt và cấu hình.	Cần cài đặt - Graylog - Elasticsearch - MongoDB Độ phức tạp trung bình.
Nền tảng hỗ trợ	Unix, Window, Linux, Ubuntu, Solaris	Linux, Solaris, Window, Unix	Window, Linux, Ubuntu

Định dạng tệp log hỗ trợ	Các loại tệp dữ liệu log phổ biến như nginx, http, database, tomcat, ...	Bất kỳ định dạng tệp nào như text, CSV, tệp log, ...	Các loại tệp dữ liệu log phổ biến như nginx, http, database, tomcat, syslog, GLEF ...
Công cụ vận chuyển dữ liệu	- Apache Kafka - RabbitMQ - Redis	Kiến trúc pipeline trong splunk	- Apache Kafka - RabbitMQ
Tổng hợp dữ liệu	Tổng hợp theo lô và Real-time	Tổng hợp theo lô và Real-time	Tổng hợp theo lô và Real-time
Khả năng tìm kiếm	Khả năng tìm kiếm và phân tích mạnh mẽ với Elasticsearch	Sử dụng ngôn ngữ tìm kiếm được xây dựng bởi Splunk áp dụng Map-Reduce	Khả năng tìm kiếm cơ bản
Khả năng xây dựng báo cáo, màn hình giám sát, trừu tượng hóa dữ liệu	Với phần mềm Kibana trong bộ giải pháp, ELK cho khả năng dựng báo cáo, xây dựng màn hình giám sát mạnh mẽ, trực quan.	Tính năng báo cáo, giám sát được xây dựng sẵn trong Splunk.	Giao diện báo cáo và giám sát đơn giản.

Ngoài ra nếu không sử dụng các giải pháp phần mềm ta cũng hoàn toàn có thể tự phát triển (code) các thành phần của một hệ thống quản lý dữ liệu log tập trung, nhưng như thế sẽ rất tốn kém cả về nguồn lực, chi phí lẫn thời gian và hiệu quả mang lại có thể là không cao.

Trong 3 giải pháp quản lý dữ liệu log tập trung được trình bày ở trên thì giải pháp Splunk tuy rất mạnh mẽ nhưng vì là phần mềm thương mại nên chi phí phải bỏ ra là rất lớn nên chúng tôi đã không lựa chọn giải pháp này. Chúng tôi đi sâu vào tìm hiểu 2 giải pháp mã nguồn mở còn lại là ELK và Graylog để chọn lựa ra được giải pháp tối ưu và phù hợp nhất với nhu cầu và hiện trạng tại Tập đoàn Bảo Việt.

**Định hướng** của chúng tôi cho bài toán quản lý dữ liệu log tập trung không chỉ đơn thuần dừng lại ở việc lưu trữ, tìm kiếm và truy vết dữ liệu log, mà còn tận dụng dữ liệu log mang tính lịch sử để kết hợp với các nguồn dữ liệu nghiệp vụ khác sử dụng trong các bài toán phân tích xu hướng, phân tích hành vi người dùng, nghiệp vụ thông minh (Business Intelligence), phân tích dữ liệu lớn (Bigdata) trong tương lai. Bên cạnh



đó với đặc thù hệ thống CNTT lớn và đa dạng nền tảng phần mềm và dữ liệu log nên giải pháp chúng tôi lựa chọn sử dụng cũng phải là giải pháp mở và mang tính tùy biến cao.

Để có thể lựa chọn được giải pháp công nghệ phù hợp, chúng tôi đã tìm hiểu về tính năng, các ưu và nhược điểm của 2 giải pháp mã nguồn mở ELK và Graylog, trong đó:

- Graylog là giải pháp được phát triển trên mục tiêu đơn thuần là lưu trữ, quản lý, cảnh báo và tìm kiếm dữ liệu log. Mọi công việc từ tổng hợp dữ liệu, cảnh báo đều có thể được thực hiện dễ dàng trên giao diện UI (User interface) trực quan nhưng khá đóng đối với người sử dụng. Graylog cung cấp các bộ trích xuất (Extractors) để trích xuất và tổng hợp các loại dữ liệu log nhất định, khi có yêu cầu cần phải trích xuất dữ liệu log phức tạp và mang tính đặc thù cao thì Graylog bắt đầu thể hiện điểm yếu so với Logstash trong bộ giải pháp ELK – Logstash cung cấp kỹ thuật trích xuất và tổng hợp dữ liệu log dựa trên công nghệ ETL (Extract, transform, load) mang tính mở, tường minh và tùy biến hơn rất nhiều. Về khả năng xây dựng báo cáo, biểu đồ, trừu tượng hóa dữ liệu trong Graylog chỉ thực hiện được ở mức cơ bản, chưa đủ linh hoạt để có thể sử dụng trong các bài toán phân tích dữ liệu, việc này ELK lại làm rất tốt với thành phần Kibana.
- ELK cũng giống Graylog là một bộ giải pháp công nghệ mã nguồn mở được sử dụng để thu thập, quản lý, phân tích dữ liệu log tập trung, tuy nhiên định hướng và mục tiêu của bộ giải pháp ELK vượt xa công việc quản lý dữ liệu log đơn thuần.


ELK gồm 3 thành phần:

- Elasticsearch: Một hệ truy hồi thông tin mạnh mẽ, được sử dụng để lưu trữ và đánh chỉ mục cho dữ liệu log.
- Logstash: Phần mềm mã nguồn mở thực hiện tiến trình đồng bộ dữ liệu ETL, thu thập dữ liệu log.
- Kibana: Phần mềm mã nguồn mở được sử dụng để trừu tượng hóa dữ liệu, xây dựng biểu đồ, màn hình giám sát và phân tích dữ liệu.

ELK so với Graylog, ban đầu sẽ khó tiếp cận và triển khai hơn, tuy nhiên giá trị giải pháp này mang lại rất đáng giá. ELK sẽ phát huy khả năng của mình nổi trội hơn so với các giải pháp khác khi cần trích xuất và tổng hợp dữ liệu đặc thù do Logstash cung cấp kỹ thuật trích xuất và tổng hợp dữ liệu rất mở và linh hoạt hay khi cần xây dựng các biểu đồ, trừu tượng hóa dữ liệu phức tạp để phân tích vấn đề - nơi mà Kibana sẽ thực hiện hoàn hảo nhiệm vụ của mình.

Bảng 1.2 : Bảng so sánh các tính năng của 2 giải pháp quản lý dữ liệu log mã nguồn mở ELK và Graylog

Tính năng	ELK Stack	Graylog
Bản quyền	Mã nguồn mở	Mã nguồn mở
Ngôn ngữ	Java, JRuby, NodeJS	Java
Độ phức tạp cài đặt	Cần cài đặt 3 thành phần: <ul style="list-style-type: none"> <li>- ElasticSearch</li> <li>- Logstash</li> <li>- Kibana</li> </ul> Độ phức tạp trung bình.	+ Cần cài đặt: <ul style="list-style-type: none"> <li>- Graylog Server</li> </ul> + Cài đặt mở rộng: <ul style="list-style-type: none"> <li>- ElasticSearch</li> <li>- MongoDB</li> </ul> Độ phức tạp trung bình.
Nền tảng hỗ trợ	Unix, Window, Linux, Ubuntu, Solaris,...	Window, Linux, Ubuntu, ...
Định dạng tệp log hỗ trợ	Các loại tệp dữ liệu log phổ biến như nginx, http, database, tomcat, ...	Các loại tệp dữ liệu log phổ biến như nginx, http, database, tomcat, syslog, GLEF ...
Công cụ làm bộ đệm dữ liệu	<ul style="list-style-type: none"> <li>- Apache Kafka</li> <li>- RabbitMQ</li> <li>- Redis</li> </ul>	<ul style="list-style-type: none"> <li>- Apache Kafka</li> <li>- RabbitMQ</li> <li>- Redis</li> </ul>
Mục đích ra đời	<ul style="list-style-type: none"> <li>- Lưu trữ, đánh chỉ mục dữ liệu.</li> <li>- Trừu tượng hóa dữ liệu phục vụ phân tích phức tạp.</li> <li>- Business Intelligence</li> <li>- Phân tích dữ liệu lớn</li> </ul>	<ul style="list-style-type: none"> <li>- Quản lý, truy vết dữ liệu log đơn thuần.</li> </ul>
Cách nhìn nhận đối với dữ liệu log	<ul style="list-style-type: none"> <li>- Coi dữ liệu log là một loại dữ liệu cần được xử lý bên cạnh các loại dữ liệu khác.</li> <li>- Coi dữ liệu log chứa các thông tin hữu ích về nhiều khía cạnh của doanh nghiệp mà cần được khai thác.</li> </ul>	<ul style="list-style-type: none"> <li>- Chỉ đơn thuần là dữ liệu log.</li> <li>- Sử dụng để tìm kiếm, truy vết.</li> </ul>

<b>Khả năng tìm kiếm</b>	<ul style="list-style-type: none"> <li>- Khả năng tìm kiếm và phân tích mạnh mẽ với ElasticSearch</li> </ul>	<ul style="list-style-type: none"> <li>- Khả năng tìm kiếm cơ bản trong Graylog</li> <li>- Nếu cài đặt mở rộng ElasticSearch sẽ tận dụng được khả năng tìm kiếm mạnh mẽ của hệ truy hỏi thông tin này</li> </ul>						
<b>Khả năng xây dựng báo cáo, màn hình giám sát, trừu tượng hóa dữ liệu</b>	<p>Với phần mềm Kibana trong bộ giải pháp, ELK cho khả năng dựng báo cáo, xây dựng màn hình giám sát mạnh mẽ, trực quan, hỗ trợ rất tốt cho bài toán phân tích dữ liệu.</p>	<p>Giao diện báo cáo và giám sát đơn giản trên Graylog</p>						
<b>Cách thức trích xuất dữ liệu</b>	<p>Logstash cung cấp khả năng trích xuất và tổng hợp dữ liệu theo công nghệ ETL mang tính mở và rất linh động.</p>	<p>Graylog cung cấp một số bộ trích xuất (extractor) cho từng loại dữ liệu log cụ thể.</p>						
<b>Khả năng tương thích với ElasticSearch</b>	<p>ElasticSearch là một thành phần trong bộ giải pháp nên bộ giải pháp luôn đồng bộ và khả năng tương thích tuyệt vời</p>	<p>Khả năng tương thích với các phiên bản ElasticSearch mới của Graylog luôn là chậm hơn</p>						
<b>Cộng đồng phát triển</b>	<p>Cộng đồng phát triển và tương tác lớn mạnh:</p> <p><a href="https://discuss.elastic.co/c/logstash">https://discuss.elastic.co/c/logstash</a></p> <ul style="list-style-type: none"> <li>- Số lượng bài đăng và tương tác cộng đồng rất lớn:</li> </ul> <p>Categories (40 more) ...</p> <table border="0"> <tr> <td>■ Elasticsear... 60.0k</td> <td>■ Logstash 21.0k</td> </tr> <tr> <td>■ Kibana 13.1k</td> <td>■ Filebeat 3.7k</td> </tr> <tr> <td>■ X-Pack 1.8k</td> <td>■ Beats 1.2k</td> </tr> </table>	■ Elasticsear... 60.0k	■ Logstash 21.0k	■ Kibana 13.1k	■ Filebeat 3.7k	■ X-Pack 1.8k	■ Beats 1.2k	<p>Cộng đồng phát triển và tương tác kém hơn rất nhiều so với ELK:</p> <p><a href="https://community.graylog.org/">https://community.graylog.org/</a></p> <ul style="list-style-type: none"> <li>- Số lượng bài đăng và tương tác trong cộng đồng còn nhiều hạn chế:</li> </ul>  <p>■ Announcements × 29 This is where we'll post important Graylog updates, such as new releases, product updates, and other useful information.</p> <p>■ Graylog × 4096 Any questions related to running Graylog, including installation and configuration, specific features, and official plugins.</p> <p>■ Development × 167 This is the place to discuss and ask questions about the development of a Graylog-related project</p>
■ Elasticsear... 60.0k	■ Logstash 21.0k							
■ Kibana 13.1k	■ Filebeat 3.7k							
■ X-Pack 1.8k	■ Beats 1.2k							

Mỗi giải pháp đều có ưu và nhược điểm riêng của chúng. Tùy thuộc vào đặc thù của doanh nghiệp, nhu cầu và định hướng xây dựng hệ thống quản lý dữ liệu log mà có sự lựa chọn giải pháp cho phù hợp. Dưới đây là bảng liệt kê các ưu/nhược điểm của 2 giải pháp quản lý dữ liệu log mã nguồn mở phổ biến ELK và Graylog.

*Bảng 1.3: Bảng ưu và nhược điểm của 2 giải pháp quản lý dữ liệu log mã nguồn mở ELK và Graylog*

	<b>ELK</b>	<b>Graylog</b>
<b>Ưu điểm</b>	Mã nguồn mở nên chi phí rẻ.	Mã nguồn mở nên chi phí rẻ.
	Hỗ trợ nhiều nền tảng hệ điều hành và loại dữ liệu log khác nhau.	Hỗ trợ nhiều nền tảng hệ điều hành và loại dữ liệu log khác nhau.
	Hỗ trợ cảnh báo tự động khi có lỗi trong dữ liệu log.	Hỗ trợ cảnh báo tự động khi có lỗi trong dữ liệu log.
	Logstash cung cấp khả năng trích xuất và tổng hợp dữ liệu mạnh mẽ và tùy biến cao phù hợp với các kiểu dữ liệu phức tạp.	Đễ dàng triển khai thu thập và giám sát trên những loại dữ liệu log thông dụng với giao diện cấu hình đồ họa thân thiện.
	Kibana cho phép xây dựng biểu đồ, trừu tượng hóa dữ liệu phục vụ các bài toán phân tích phức tạp.	
	Tương thích hoàn toàn với Elasticsearch cho phép kiểm soát việc đánh chỉ mục tốt hơn trên Elasticsearch.	
	Sử dụng dữ liệu log không chỉ để tìm kiếm, cảnh báo, truy vết. Bên cạnh đó còn có thể phân tích, trừu tượng hóa dữ liệu phức tạp để tìm được các thông tin hữu ích.	
	Hệ thống mở, mang tính tùy biến cao.	
	Cộng đồng phát triển và tương tác rất lớn.	

<b>Nhược điểm</b>	Triển khai cần nhiều kiến thức cấu hình hệ thống hơn, không hỗ trợ giao diện đồ họa mạnh như Graylog.	Công cụ trích xuất dữ liệu của Graylog cung cấp còn sơ sài, không có tính tùy biến cao như trên Logstash của giải pháp ELK.
		Chức năng xây dựng biểu đồ đơn giản, khả năng trừu tượng hóa dữ liệu kém, không phù hợp để sử dụng cho các bài toán phân tích dữ liệu, BI, BigData.
		Khả năng tương thích với các phiên bản ElasticSearch mới chậm. Ít linh hoạt hơn trong vấn đề tạo chỉ mục trên ElasticSearch.
		Coi dữ liệu log chỉ đơn thuần là dữ liệu log phục vụ tìm kiếm, cảnh báo lỗi và truy vết, không thể phân tích các trường hợp phức tạp.
		Hệ thống có khả năng tùy biến kém hơn ELK.
		Cộng đồng phát triển, tương tác và chia sẻ còn hạn chế.

Sau khi cân nhắc về tính năng, ưu nhược điểm, chi phí đối với từng giải pháp và định hướng ban đầu khi xây dựng hệ thống quản lý dữ liệu log tập trung, chúng tôi quyết định lựa chọn nền tảng công nghệ ELK làm giải pháp công nghệ cho bài toán quản lý và phân tích dữ liệu log do ELK hội tụ một số ưu điểm như:

- Mã nguồn mở nên chi phí rẻ, cộng đồng phát triển lớn mạnh.
- Hỗ trợ nhiều nền tảng hệ điều hành và loại dữ liệu log khác nhau, rất phù hợp với hiện trạng tại Bảo Việt.
- Nền tảng tìm kiếm ElasticSearch rất mạnh mẽ, dễ dàng mở rộng theo chiều ngang nếu hệ thống tăng trưởng lớn, tích hợp được sâu rộng với các loại công cụ, ngôn ngữ phân tích dữ liệu lớn như R, Python, Machine learning.

- Khả năng xây dựng báo cáo, phân tích và trừu tượng hóa dữ liệu mạnh mẽ với Kibana phù hợp với các bài toán phân tích dữ liệu lớn, phân tích hành vi người dùng, Business Intelligence.
- Khả năng trích xuất và tổng hợp dữ liệu log mang tính tùy biến cao, phù hợp sử dụng đối với những trường hợp dữ liệu log phức tạp.
- Giải pháp có tính mở và hỗ trợ tùy biến tốt.

## 1.4. Tìm hiểu nền tảng công nghệ ELK

### 1.4.1. Giới thiệu ELK

ELK là một bộ giải pháp công nghệ mã nguồn mở được sử dụng để thu thập, quản lý, phân tích dữ liệu log tập trung.

ELK gồm 3 thành phần:

- ElasticSearch: Một hệ truy hồi thông tin mạnh mẽ, được sử dụng để lưu trữ và đánh chỉ mục cho dữ liệu log.
- Logstash: Phần mềm mã nguồn mở thực hiện tiến trình đồng bộ dữ liệu ETL, thu thập dữ liệu log, chuyển đổi, làm sạch và đưa vào lưu trữ trong ElasticSearch để đánh chỉ mục.
- Kibana: Phần mềm mã nguồn mở được sử dụng để trừu tượng hóa dữ liệu, xây dựng biểu đồ, màn hình giám sát và phân tích dữ liệu.

### 1.4.2. ElasticSearch

#### 1.4.2.1. Giới thiệu ElasticSearch

ElasticSearch là một giải pháp truy hồi thông tin và phân tích dữ liệu phân tán mã nguồn mở mạnh mẽ và có tính mở rộng cao. ElasticSearch được phát triển trên nền tảng thư viện search-engine mã nguồn mở nổi tiếng “*Apache Lucene*”. Apache Lucene sử dụng ngôn ngữ Java và khá phức tạp để sử dụng, ElasticSearch kế thừa Apache Lucene và che dấu sự phức tạp của Lucene đằng sau các RESTful API. ElasticSearch cho phép lưu trữ, tìm kiếm, và phân tích lượng lớn dữ liệu thời gian thực. Nó thường được sử dụng để hỗ trợ cho các ứng dụng có nhu cầu tìm kiếm phức tạp, cần tốc độ nhanh và các ứng dụng phân tích dữ liệu lớn.

Một số bài toán có thể sử dụng của ElasticSearch:

- Tìm kiếm sản phẩm trên trang web bán hàng.
- Thu thập log hệ thống, dữ liệu giao dịch phục vụ phân tích, tìm kiếm. Với bài toán này thì bên cạnh ElasticSearch cần sử dụng thêm 2 công cụ là Logstash và Kibana. Ba phần mềm mã nguồn mở này hợp thành bộ giải pháp ELK (ElasticSearch, Logstash và Kibana).
- Phân tích dữ liệu lớn (BigData analytic). Với bài toán này, ElasticSearch sẽ đóng vai trò là nơi lưu trữ dữ liệu từ nhiều

nguồn, nhiều loại dữ liệu khác nhau được tích hợp về bởi các công cụ ETL, Streaming. ElasticSearch có thể kết hợp sử dụng với các công cụ phân tích dữ liệu lớn mạnh mẽ như Spark, R hay Python giúp cho các nhà khoa học dữ liệu có thể phân tích dữ liệu một cách thời gian thực để đưa ra các thông tin hữu ích.

- Các bài toán Business Intelligent Analytic để đánh giá nhanh, phân tích nhanh trên một lượng lớn dữ liệu để đưa ra các quyết định kinh doanh kịp thời. Sử dụng bộ giải pháp ELK (ElasticSearch, Logstash và Kibana) để tập hợp dữ liệu, xây dựng các màn hình điều khiển (dardboard) để có thể phân tích được số liệu rất trực quan.

Một số tổ chức lớn sử dụng ElasticSearch:

- Trang Wikipedia sử dụng ElasticSearch để cung cấp máy tìm kiếm toàn văn (full-text search) với kết quả tìm kiếm được tô sáng.
- Trang The Guardian sử dụng ElasticSearch để kết hợp dữ liệu của người đọc với dữ liệu mạng xã hội để cung cấp các hồi đáp thời gian thực giúp tăng trải nghiệm người dùng.
- Trang cộng đồng cho các nhà phát triển phần mềm nổi tiếng Stack Overflow sử dụng ElasticSearch làm máy tìm kiếm toàn văn kết hợp với vị trí địa lý của người dùng để đưa ra các kết quả tìm kiếm chính xác nhất cho câu truy vấn của người dùng.
- Trang quản lý mã nguồn mở nổi tiếng GitHub sử dụng ElasticSearch để quản lý hơn 130 triệu dòng codes.

#### 1.4.2.2. Kiến trúc ElasticSearch

##### a. ElasticSearch Cluster

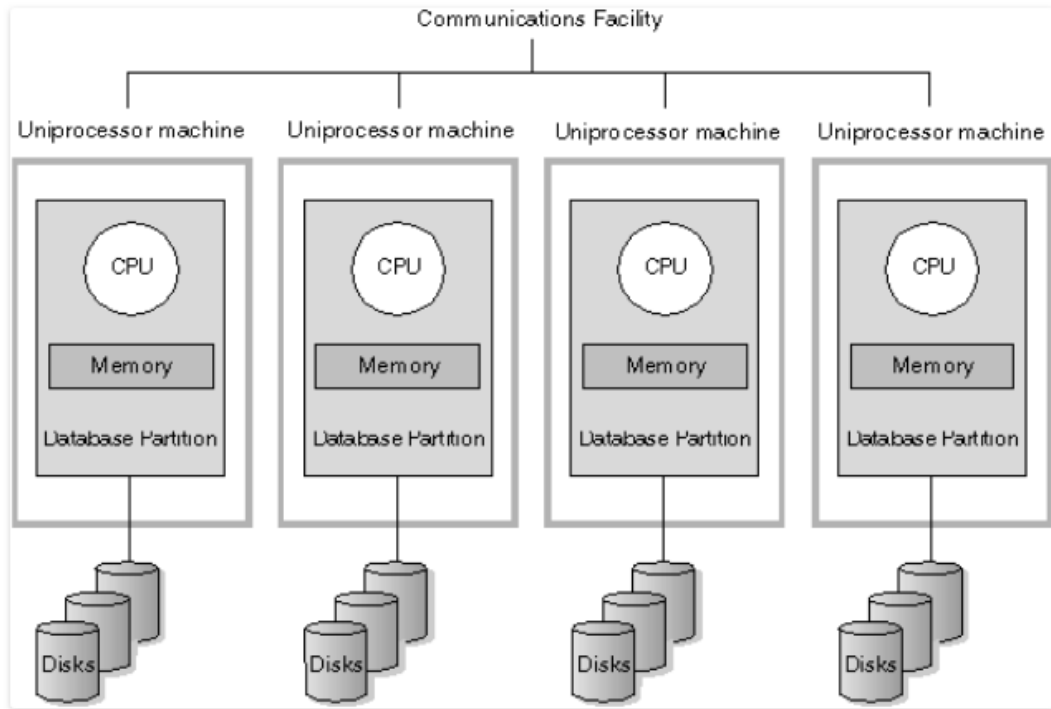
ElasticSearch Cluster được xây dựng theo ý tưởng kiến trúc MPP (Massive Parallel Processing).

Massive Parallel Processing là một hệ thống gồm nhiều nút (node), hoạt động cùng nhau để cùng thực hiện một chương trình, trong đó mỗi node sẽ xử lý một phần riêng của chương trình trên chính tài nguyên của node đó (memory, CPU, ...). Do đó mà một hệ thống MPP còn được gọi là hệ thống “Shared nothing” (vì bản chất các node trong cụm không chia sẻ tài nguyên gì để tính toán, chúng xử lý dữ liệu riêng của chúng trên sức mạnh tài nguyên riêng của chúng).

Để có thể xử lý lượng dữ liệu khổng lồ, dữ liệu trong giải pháp MPP thường được phân chia giữa các node thành các phân đoạn (shard), mỗi nút sẽ xử lý dữ liệu cục bộ của nó. Điều này càng tăng tốc độ xử lý dữ liệu, bởi vì sử dụng lưu trữ chia sẻ cho giải pháp MPP sẽ là một khoản đầu tư lớn hơn, phức tạp hơn, tốn kém

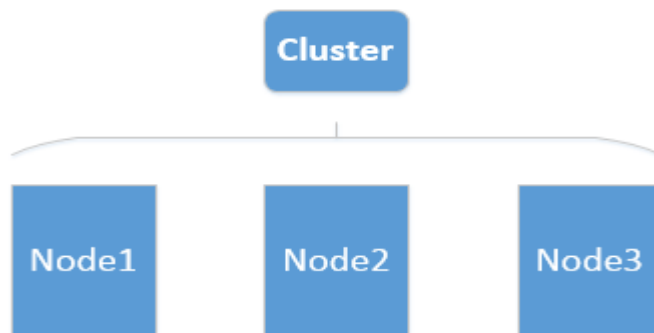
hơn, ít khả năng mở rộng hơn, sử dụng lưu lượng mạng cao hơn và ít tính toán song song hơn.

Với cách thiết kế này, một hệ thống MPP sẽ rất dễ dàng để mở rộng, ta chỉ cần thêm node vào cụm cluster theo chiều ngang là có thể mở rộng năng lực tính toán cho toàn cụm. Mô hình MPP như sau:



Hình 1.2 : Mô hình Massive Parallel Processing

Một Cụm ElasticSearch Cluster bao gồm một hoặc nhiều nút (nodes) có cùng tên Cluster mà nó tham gia vào. Các nodes trong cụm Cluster làm việc cùng nhau và chia sẻ dữ liệu và tải (workload) với nhau. Khi một node được thêm vào hoặc rời khỏi cụm thì Cluster tự động tổ chức và tính toán lại dữ liệu và năng lực tính toán.



Hình 1.3 : Mô hình cụm Cluster của ElasticSearch



Một node là một máy chủ riêng lẻ, là một phần của cụm Cluster, tham gia vào quá trình đánh chỉ mục và tìm kiếm của cụm Cluster. Cũng giống như Cluster, mỗi node được định danh bởi một tên duy nhất và được sinh ngẫu nhiên tại thời điểm khởi động hệ thống. Tất nhiên chúng ta có thể chỉ định tên cho các node này cho mục đích quản lý. Mỗi node có thể tham gia (join) vào một cluster mặc định là “elasticsearch” Cluster nếu không được chỉ định.

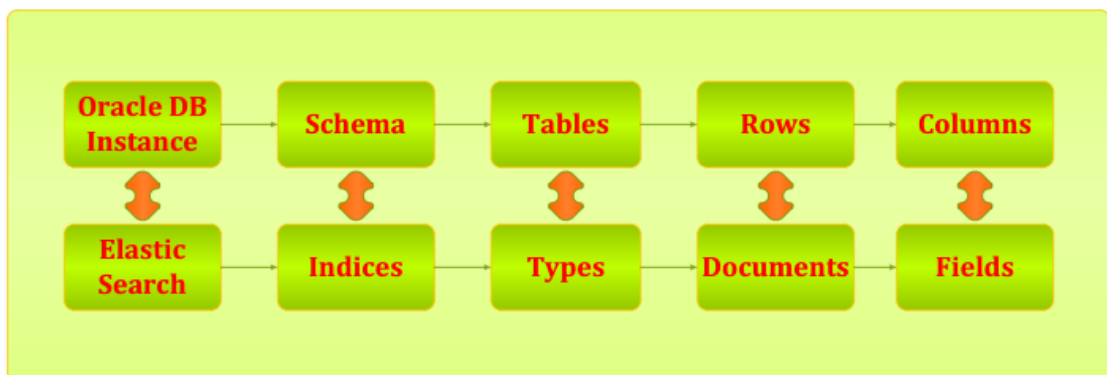
Mỗi cụm Elasticsearch Cluster có một Master node chịu trách nhiệm quản lý các thay đổi trong toàn cụm như tạo, xóa index hoặc thêm hay loại bỏ node vào/ra cụm Cluster. Master node sẽ không tham gia vào quá trình xử lý tìm kiếm. Bất kỳ node nào trong cụm cũng có thể trở thành master node. Với cụm Cluster chỉ có 1 node thì node đó sẽ thực hiện cả vai trò là master node và node xử lý tìm kiếm.

Tất cả các nodes trong cụm Cluster được kết nối, chia sẻ dữ liệu và tải với nhau. Tất cả các nodes đều biết chính xác dữ liệu được lưu trữ ở đâu, do đó khi có một yêu cầu cần xử lý được gửi đến chúng có thể trực tiếp xử lý hoặc chuyển tiếp yêu cầu đến node mà đang thực sự chứa dữ liệu cần xử lý và trả ra kết quả.

### b. Các khái niệm cơ bản trong Elasticsearch

ElasticSearch được cộng đồng mã nguồn mở phát triển đã trải qua rất nhiều phiên bản, phiên bản chính thức mới nhất tại thời điểm chúng tôi nghiên cứu là phiên bản 6.6. Trong phiên bản này có sự thay đổi so với các phiên bản 5.x trước đó, đó là chỉ có một kiểu mapping type duy nhất trong một Elasticsearch Index. Để hiểu rõ hơn chúng ta sẽ nghiên cứu các thành phần trong Elasticsearch.

Các thành phần trong Elasticsearch khi so sánh với một cơ sở dữ liệu quan hệ (Ví dụ cơ sở dữ liệu Oracle) như sau:



Hình 1.4 : So sánh các thành phần của Elasticsearch với Cơ sở dữ liệu quan hệ

Trong đó: Chỉ mục (Index) trong Elasticsearch được coi tương đương với Lược đồ (Schema), Kiểu (Type) tương đương với khái niệm bảng (Table), Tài liệu (Document)

tương đương với bản ghi (Row) và Trường (Field) tương đương với Cột (Column) trong cơ sở dữ liệu quan hệ.

### ➤ **Index**

ElasticSearch sử dụng chỉ mục ngược để đánh chỉ mục cho các tài liệu. Một chỉ mục trong ElasticSearch là một khái niệm logic, nó bao gồm tập hợp các tài liệu có một số đặc điểm tương tự nhau. Ví dụ: một chỉ mục cho dữ liệu khách hàng, một chỉ mục khác cho danh mục sản phẩm và một chỉ mục khác cho dữ liệu đơn hàng. Một chỉ mục được xác định bằng một tên duy nhất (phải là chữ thường) và tên này được sử dụng khi thực hiện các hoạt động như lập chỉ mục, tìm kiếm, cập nhật và xóa đối với các tài liệu trong chỉ mục đó.

### ➤ **Type**

Type đại diện cho kiểu của tài liệu hay thực thể được đánh chỉ mục. Một loại (Type) là một danh mục / phân vùng logic của chỉ mục để cho phép lưu trữ các loại tài liệu khác nhau trong cùng một chỉ mục, ví dụ: Index có tên là twitter có một loại cho người dùng (user type), một loại khác cho bài viết trên blog (tweet type).

### ➤ **Document**

Tài liệu (document) là một đơn vị thông tin cơ bản có thể đánh chỉ mục. Document giống như row của table trong cơ sở dữ liệu quan hệ. Thuật ngữ “document” trong ElasticSearch chỉ đến các tài liệu được thể hiện dưới dạng JSON.

Hầu hết các objects và documents đều có thể được thể hiện dưới dạng JSON document với key và value. Một key là tên của một field (hay property), và một value có thể là một kiểu String, Boolean, Integer, ...

Ví dụ một document được thể hiện dưới dạng JSON document:

```

{
  "name":      "John Smith",
  "age":      42,
  "confirmed": true,
  "join_date": "2014-06-01",
  "home": {
    "lat":    51.5,
    "lon":    0.1
  },
  "accounts": [
    {
      "type": "facebook",
      "id":   "johnsmith"
    },
    {
      "type": "twitter",
      "id":   "johnsmith"
    }
  ]
}

```

Một tài liệu không chỉ chứa dữ liệu, nó còn có siêu dữ liệu (metadata) – thông tin về tài liệu đó, bao gồm:

- Index: Nơi mà tài liệu được đánh chỉ mục và lưu trữ (Tên index)
- Type: Lớp/Kiểu mà document thể hiện. (VD: kiểu “doc”, kiểu “blog” hay “comment”, ...)
- Id: Định danh duy nhất cho tài liệu để phân biệt giữa tài liệu này với tài liệu khác.

#### ➤ Field

Khái niệm Field để chỉ một trường của tài liệu JSON. Field trong tài liệu JSON được biểu diễn dưới dạng <Key, Value>, với Key là tên của Field và Value là giá trị của Field đó, Value có thể là kiểu String, Boolean, Integer, ...

#### ➤ Shard

Khi 1 chỉ mục quá lớn, không thể lưu trữ trên 1 node thì Elasticsearch cho phép chia chỉ mục đó ra thành các phân đoạn (shards). Index chỉ có thể được chia thành các shards khi cụm Cluster có 2 nodes trở lên. Việc chia Index thành các shards có các lợi ích sau:

- Hệ thống có thể mở rộng theo chiều ngang.
- Cho phép tìm kiếm song song (parallel) trên các shards.

ElasticSearch khuyến nghị phải tạo 1 hoặc nhiều bản sao cho mỗi shard của index, bản sao này gọi là replica shard. Bản shard gốc gọi là primary/original shard. Việc tạo nhiều replica shard cho phép tìm kiếm song song,

tăng hiệu năng tìm kiếm. Số Primary Shards và số Replica Shards có thể thiết đặt khi tạo index. Sau khi Index được tạo thì số Replica Shards có thể thay đổi được, nhưng số Primary Shards là không thể thay đổi. Mặc định, nếu cụm Cluster có từ 2 nodes trở lên mà khi tạo index không có chỉ định rõ số Primary Shards và số Replica Shards thì Elasticsearch mặc định số Primary Shards là 5 và mỗi Primary Shard có 1 Replica Shard. Mỗi Shard trong Elasticsearch là một Lucene Index. Trong 1 Lucene Index thì số lượng documents tối đa có thể chứa được là  $2^{31}$ . Khi cụm cluster được mở rộng (thêm node) hoặc co lại (loại bỏ node) thì cluster sẽ tự động tính toán và di chuyển các shards qua lại các nodes để cụm Cluster đạt trạng thái cân bằng (balanced).

#### 1.4.2.3. Mô hình truy hồi thông tin của Elasticsearch

Mô hình truy hồi thông tin mặc định mà Elasticsearch sử dụng là mô hình **BM25**. Đây là thuật toán mặc định được sử dụng bởi Apache Lucene và Elasticsearch phiên bản 6.6. BM25 có nguồn gốc từ mô hình liên quan xác suất (Probabilistic relevance model).

BM25 so với các mô hình TF/IDF và VSM (Vector Space Model) có những điểm tương đồng nhất định, tuy nhiên vẫn có những điểm khác nhau giữa các mô hình này. Cả 3 mô hình đều sử dụng các thành phần: term frequency (TF), inverse document frequency (IDF) và field length norm để tính toán trọng số liên quan của tài liệu đến câu truy vấn. Nhưng việc sử dụng 3 thành phần này giữa các mô hình là có sự khác nhau tạo nên sự khác biệt của riêng chúng. Định nghĩa của 3 thành phần này như sau:

- Term frequency (TF)	Độ thường xuyên xuất hiện của từ tố (term) trong một tài liệu (hoặc field trong ngữ cảnh của Elasticsearch). Term càng xuất hiện thường xuyên thì càng tăng độ liên quan của tài liệu với term đó.
- Inverse document frequency (IDF)	Độ xuất hiện thường xuyên của term trong nhiều văn bản. Term càng xuất hiện nhiều trong nhiều văn bản (term phổ biến) thì có trọng số càng thấp, ngược lại term càng xuất hiện ít trong nhiều văn bản (term không phổ biến) thì trọng số càng cao. Thuật toán này để giảm mức độ quan trọng của term khi nó xuất hiện trong nhiều văn bản, nó được sử dụng để loại trừ các term phổ biến, lọc lấy các term đặc trưng của văn bản.

- Field length norm (FLN)	Độ dài của trường tài liệu. Khi một từ xuất hiện trong một trường dữ liệu ngắn, nhiều khả năng nó mang đặc trưng cho đoạn văn bản đó cao hơn là khi từ đó xuất hiện trong đoạn văn bản dài hơn. Do đó, trường càng ngắn thì trọng lượng càng cao hơn trường dài.
---------------------------	--

Cả 2 mô hình BM25 và TF/IDF đều sử dụng IDF để phân biệt giữa các từ phổ biến (trọng số thấp) và các từ không phổ biến (trọng số cao). Cả 2 mô hình đều chấp nhận rằng khi một term xuất hiện thường xuyên trong một tài liệu thì tài liệu đó càng có sự liên quan đến term (term frequency).

Công thức tính trọng số của mô hình BM25 cho tài liệu D được trả ra cho câu query Q như sau:

$$\text{score}(D, Q) = \sum_{i=1}^n \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}, \quad (1)$$

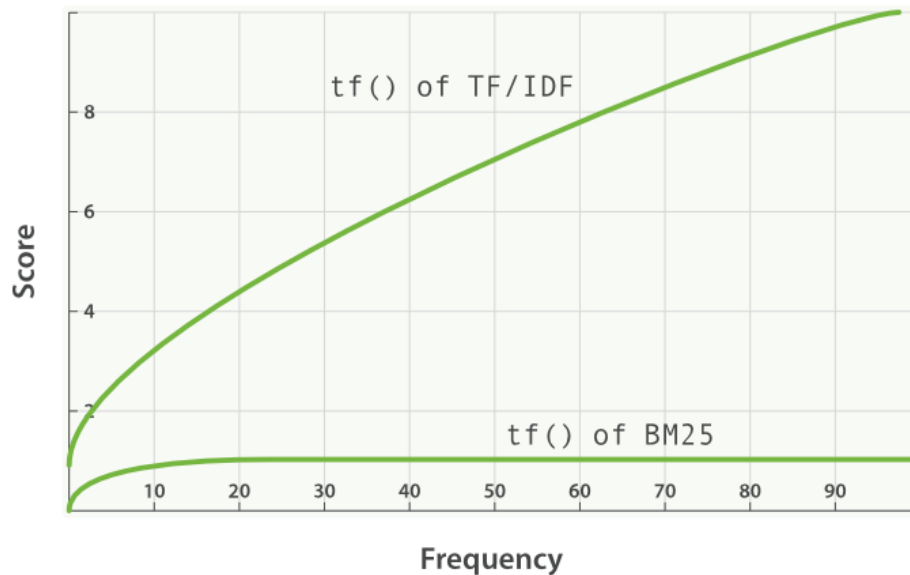
Trong đó:

- Q: Câu truy vấn
- D: Tài liệu trả ra cho câu truy vấn
- $k_1$ : Ngưỡng bão hòa cho Term frequency, mặc định là 1.2
- $b$ : Ngưỡng bão hòa cho Field length norm, mặc định là 0.75
- $|D|$ : Độ dài của văn bản D
- avgdl: Độ dài trung bình của văn bản trong tập văn bản
- $\text{IDF}(q_i)$ : Là Inverse document frequency của từ  $q_i$  trong tập văn bản
- $f(q_i, D)$ : Là term frequency của từ  $q_i$  trong tài liệu D

Trong thực tế nếu một từ xuất hiện nhiều trong 1 tài liệu thì nó cũng xuất hiện trong nhiều tài liệu khác, các từ đó được gọi là các từ dừng (Stopwords). Mô hình TF/IDF được thiết kế để loại trừ các từ phổ biến (hay từ dừng – stop word). Mô hình TF/IDF ra đời khi mà dung lượng bộ nhớ còn hạn chế và đắt đỏ, nên để lựa chọn giữa tối ưu hiệu năng hay tối ưu độ chính xác thì TF/IDF đã lựa chọn tối ưu hiệu năng bằng cách loại bỏ bớt các Stop Word, kết hợp với các giải thuật nén, khi đó kích thước chỉ mục (index) được giảm và làm tăng tốc độ truy vấn.

Điều gì xảy ra khi tính toán trọng số cho các từ dừng trong tài liệu? Đó là trọng số của từ dừng sẽ ngày càng cao khi tần xuất xuất hiện của từ đó trong tài liệu tăng lên, trong khi thực tế các từ dừng sẽ không phải là các từ đặc trưng của tài liệu và không chứa nhiều thông tin cho tài liệu, đó là một trọng số ảo. Mô hình TF/IDF loại trừ việc đó bằng cách loại bỏ các từ dừng khi thực hiện tạo chỉ mục (index). Trọng số của từ trong mô hình TF/IDF tỉ lệ tuyến tính với tần xuất xuất hiện của từ trong tài liệu. Tuy nhiên nhược điểm của mô hình TF/IDF khi loại bỏ các từ dừng là không thể tìm kiếm chính xác cụm văn bản và bị hạn chế tìm kiếm về mặt ngữ nghĩa.

Trong mô hình BM25, với việc thiết lập các tham số ngưỡng bão hòa  $k_1$  (mặc định 1.2) và  $b$  (mặc định 0.75) thì trọng số của từ tổ không tuyến tính với tần xuất xuất hiện của từ tổ đó trong tài liệu như với mô hình TF/IDF truyền thống. Ví dụ, với từ xuất hiện từ 5 đến 10 lần trong tài liệu sẽ có trọng số cao hơn các từ chỉ xuất hiện 1 đến 2 lần. Trong khi đó từ xuất hiện 20 lần với từ xuất hiện hàng nghìn lần đều có trọng số ngang nhau (ngưỡng bão hòa). Cụ thể được thể hiện như hình dưới:



Hình 1.5 : Biểu đồ Term Frequency của mô hình BM25 và TF/IDF

#### 1.4.2.4. Tìm kiếm trong Elasticsearch

Có 2 dạng tìm kiếm trong Elasticsearch là lọc (**Filter**) và truy vấn (**Query**). Sự khác nhau giữa 2 dạng tìm kiếm này là Query sẽ tính toán độ liên quan và xếp hạng kết quả tìm kiếm; trong khi đó filter sẽ trả ra kết quả chính xác như điều kiện tìm kiếm và không tính toán, xếp hạng kết quả.

##### a. Filter

Khi muốn tìm kiếm chính xác các tài liệu chứa một giá trị nào đó ta sẽ sử dụng câu lệnh lọc (filter). Bởi vì Filter không tính toán độ liên quan và xếp hạng kết quả tìm kiếm nên tốc độ của filter là rất nhanh. Và để tăng tốc hơn nữa cho các câu lệnh filter, Elasticsearch hỗ trợ lưu vào bộ nhớ đệm (Cache) các kết quả tìm kiếm của

câu lệnh Filter phục vụ cho các lần tìm kiếm sau. Ví dụ câu lệnh Filter trong ElasticSearch:

```
GET /my_store/products/_search
{
  "query" : {
    "filter" : {
      "term" : {
        "price" : 20
      } } } }
```

Lệnh filter trên tương đương với câu lệnh truy vấn SQL trong cơ sở dữ liệu quan hệ sau:

```
SELECT document
FROM products
WHERE price = 20
```

Kết quả trả ra cho câu tìm kiếm chính xác (filter) tùy thuộc vào kiểu dữ liệu của từ tố và kiểu của trường trong chỉ mục. Đối với câu filter cho kiểu dữ liệu number, bool hoặc date sẽ luôn cho ra kết quả chính xác với điều kiện tìm kiếm. Tuy nhiên với kiểu dữ liệu String (chuỗi ký tự) thì kết quả tìm kiếm phục thuộc vào cách đánh chỉ mục cho trường dữ liệu này. Trong ElasticSearch có 2 kiểu đánh chỉ mục cho trường dữ liệu String là kiểu Keyword và kiểu Text. Trong đó với kiểu Keyword ElasticSearch sẽ không sử dụng tiến trình phân tích từ tố (analysis) trong quá trình đánh chỉ mục, ngược lại đối với kiểu Text, ElasticSearch sẽ sử dụng tiến trình analysis khi đánh chỉ mục. Ví dụ tìm kiếm lọc (filter) giá trị chính xác cho dữ liệu kiểu chuỗi ký tự:

```
GET /my_store/products/_search
{
  "query" : {
    "filtered" : {
      "filter" : {
        "term" : {
          "productID" : " Dai Hoc Cong Nghe"
        } } } }
}
```

Câu truy vấn trên tương đương với câu SQL sau trong cơ sở dữ liệu quan hệ:

```
SELECT product
FROM products
WHERE productID = "Dai Hoc Cong Nghe"
```

Khi đó, nếu trường “productID” được đánh chỉ mục theo kiểu Keyword thì câu tìm kiếm chính xác trên sẽ cho ra kết quả. Ngược lại, nếu trường “productID” được đánh chỉ mục theo kiểu Text thì câu tìm kiếm sẽ không có kết quả trả ra, nguyên nhân là do khi đánh chỉ mục kiểu Text, Elasticsearch sẽ sử dụng tiến trình Analysis để phân tích từ tổ trong trường dữ liệu “productID” và đánh chỉ mục ngược cho riêng từng từ tổ.

### b. Query

Khác với câu lệnh Filter thì câu lệnh Query sẽ tính toán độ liên quan và xếp hạng kết quả trả ra cho câu truy vấn. Có 2 khía cạnh quan trọng nhất của câu lệnh truy vấn là:

- Độ liên quan (Relevance): Tính toán và xếp hạng độ liên quan của kết quả trả ra cho câu truy vấn. Thuật toán để tính toán mặc định trong Elasticsearch là thuật toán BM25.
- Phân tích từ tổ (Analysis): Tiến trình phân tích văn bản thành các từ tổ (token) để phục vụ đánh chỉ mục ngược và truy vấn trong chỉ mục ngược.

Không phải câu truy vấn nào cũng có đầy đủ 2 khía cạnh trên. Elasticsearch bao gồm 2 loại câu truy vấn là truy vấn theo từ tổ (term query) và truy vấn toàn văn (match query hay full-text query).

#### ➤ Truy vấn từ tổ (term query)

Truy vấn theo từ tổ là tìm kiếm các tài liệu có chứa chính xác từ tổ cần tìm kiếm bên trong chỉ mục ngược. Câu truy vấn theo từ tổ không chứa quá trình phân tích từ tổ (analysis).

Ví dụ câu truy vấn theo từ tổ trong Elasticsearch:

```
POST _search
{
  "query": {
    "term": { "user": "Kimchy" }
  }
}
```



Về cách thức hoạt động, câu truy vấn theo từ tổ hoàn toàn giống với câu tìm kiếm chính xác (Filter), chỉ khác rằng câu truy vấn theo từ tổ có tính toán và xếp hạng kết quả trả ra còn câu Filter thì không.

### ➤ Truy vấn toàn văn (Full-text Search)

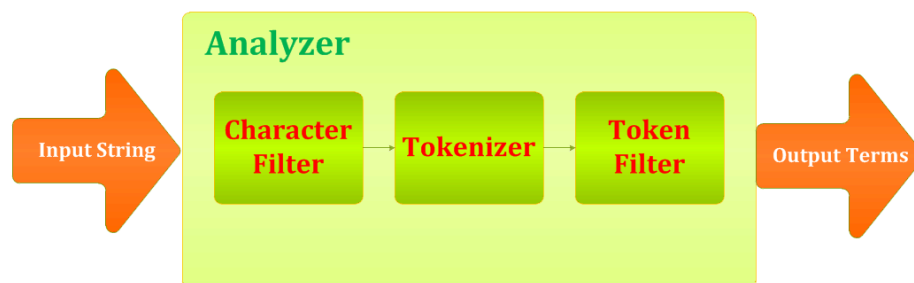
Truy vấn toàn văn trong Elasticsearch là câu truy vấn bao gồm đầy đủ cả 2 tiến trình phân tích từ tổ (Analysis) và tính toán độ liên quan của kết quả trả về cho câu truy vấn (Relevance). Với truy vấn toàn văn, để kết quả được trả ra thực sự chính xác, cần đảm bảo tiến trình phân tích được áp dụng trong quá trình đánh chỉ mục và trong quá trình truy vấn phải tương tự nhau. Ví dụ câu truy vấn toàn văn:

```
GET /my_index/my_type/_search
{
  "query": {
    "match": {
      "title": "Dai hoc cong nghe"
    }
  }
}
```

### ➤ Tiến trình phân tích (Analysis)

Tiến trình phân tích từ tổ có nhiệm vụ phân tách trường dữ liệu chuỗi thành các từ tổ được chuẩn hóa để phục vụ việc đánh chỉ mục và tìm kiếm toàn văn. Tiến trình phân tích được thực hiện bởi Analyzer, có nhiều loại analyzer trong Elasticsearch, mặc định là “English Analyzer”.

Analyzer gồm 3 module: Character filters, tokenizers, và token filters



Hình 1.6 : Tiến trình phân tích từ tổ (Analysis) trong Elasticsearch

Trong đó:

- **Character filter** : nhận chuỗi ký tự gốc đầu vào, sau đó có thể thêm, xóa, thay đổi các ký tự cho phù hợp trước khi đưa vào module

“tokenizers”, một analyzer có thể không có hoặc có nhiều Character filters.

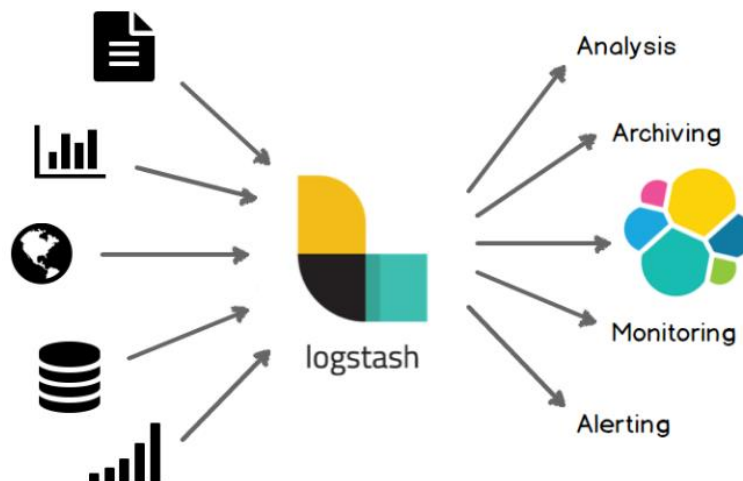
- **Tokenizer** : nhận chuỗi các ký tự đầu vào, chia chuỗi ký tự thành các từ tổ riêng lẻ (thường là các từ “word” riêng lẻ), và cho đầu ra là một chuỗi các từ tổ. Ví dụ: “Whitespace tokenizer” sẽ thực hiện phân tách các từ tổ dựa trên khoảng trắng. Tokenizer cũng làm nhiệm vụ đánh dấu thứ tự, vị trí của từng từ tổ (được sử dụng cho các truy vấn từ và cụm từ). Một Analyzer bắt buộc phải có một Tokenizer.

- **Token filter**: nhận một luồng các từ tổ (đầu ra từ Tokenizer), và nó có thể thêm, xóa, sửa các từ tổ cho phù hợp để được đầu ra là các từ tổ được chuẩn hóa phục vụ quá trình đánh chỉ mục ngược. Ví dụ: “lowercase Token filter” sẽ chuyển toàn bộ các từ tổ sang dạng viết thường (lowercase), hoặc một “stop token filter” sẽ loại bỏ toàn bộ các từ dừng (stop words). Một Analyzer có thể không có hoặc có nhiều token filter, nếu có nhiều token filter, chúng sẽ được thực hiện theo thứ tự.

### 1.4.3. Logstash

#### 1.4.3.1. Giới thiệu LogStash

LogStash là phần mềm thu thập dữ liệu mã nguồn mở được viết trên nền tảng Java với khả năng thu thập dữ liệu thời gian thực (realtime). LogStash có khả năng tự động thu thập dữ liệu từ các nguồn khác nhau, sau đó biến đổi, chuẩn hóa dữ liệu phù hợp với nơi sẽ lưu trữ nó. LogStash còn sử dụng để làm sạch dữ liệu phục vụ cho các bài toán phân tích và trực quan hóa dữ liệu.



Hình 1.7 : Giới thiệu logstash

LogStash ban đầu ra đời với mục đích để tổng hợp dữ liệu log của hệ thống phục vụ quản lý dữ liệu log tập trung, tuy nhiên sau đó khả năng của nó đã vượt xa kỳ vọng. LogStash còn có thể được sử dụng để tổng hợp và biến đổi đa dạng các kiểu dữ liệu từ

các nguồn khác nhau phục vụ rất nhiều bài toán khác nhau như cảnh báo, giám sát, lưu trữ, phân tích và trừu tượng hóa dữ liệu.

#### **1.4.3.2. Cơ sở lý thuyết công nghệ tích hợp dữ liệu ETL**

Công nghệ ETL là công nghệ sử dụng kỹ thuật hợp nhất dữ liệu, cho phép kết xuất dữ liệu từ các cơ sở dữ liệu nguồn, chuyển đổi dữ liệu đó thành dữ liệu phù hợp với yêu cầu nghiệp vụ từ đó đưa dữ liệu này vào cơ sở dữ liệu đích.

Dữ liệu có thể được kết xuất theo cơ chế pull và push. Chế độ pull thường được sử dụng trong các ứng dụng chạy ngầm (batch job) và thực hiện theo thời gian đã ấn định trước. Chế độ push thường được sử dụng trong các ứng dụng tích hợp trực tuyến và thực hiện khi có các sự kiện thay đổi dữ liệu phát sinh.

Công việc thực hiện trong ETL được mô tả trong ba bước chính sau:

➤ **Bước 1 : Kết xuất dữ liệu (Extract)**

Kết xuất dữ liệu từ các nguồn dữ liệu. Các nguồn dữ liệu thường khác nhau cả về cấu trúc và thường không đồng nhất nên cần chú ý chọn sản phẩm tích hợp có hỗ trợ nguồn dữ liệu mong muốn.

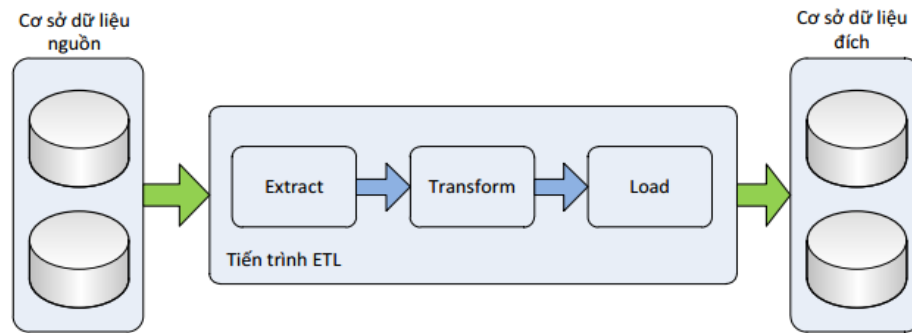
➤ **Bước 2: Chuyển đổi dữ liệu (Transform)**

Tại bước này các công đoạn sau có thể được sử dụng:

- Làm sạch dữ liệu (Ví dụ: đổi giá trị bị thiếu null thành giá trị mặc định, chuẩn hóa dữ liệu Nam là 0 và Nữ là 1...).
- Lọc dữ liệu : Lựa chọn các trường dữ liệu để xử lý, các bản ghi dữ liệu sẽ xử lý.
- Chia nhỏ dữ liệu : Chia một trường dữ liệu trong dữ liệu nguồn ra các trường nhỏ hơn.
- Hợp nhất dữ liệu từ các dữ liệu đã lấy ở bước 1.
- Loại bỏ những dữ liệu không đủ điều kiện để đưa vào dữ liệu đích.

➤ **Bước 3. Đưa dữ liệu đã được xử lý vào cơ sở dữ liệu đích**

Các bước xử lý được thể hiện qua hình vẽ dưới đây:

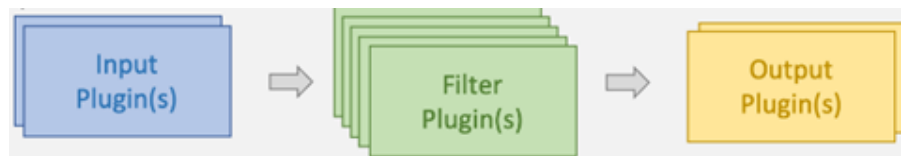


Hình 1.8 : Tiến trình ETL

### 1.4.3.3. Mô hình hoạt động của LogStash

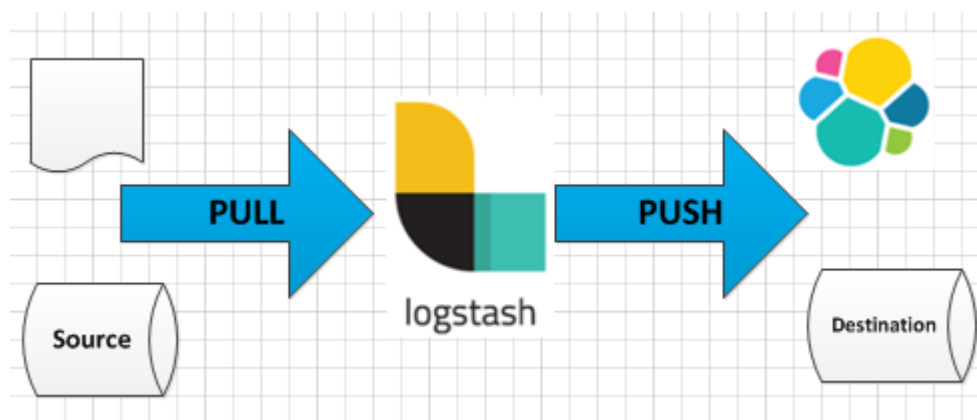
LogStash hoạt động dựa trên công nghệ tích hợp dữ liệu ETL với nền tảng là kỹ thuật hợp nhất dữ liệu (Data Consolidation) để tích hợp dữ liệu từ nhiều nguồn khác nhau về nơi lưu trữ phục vụ các bài toán khác nhau.

Trong đó, 3 tiến trình của công nghệ ETL là Extract, Transform và Load tương ứng với 3 thành phần trong mô hình triển khai LogStash gồm: Input Plugin, Filter Plugin và Output Plugin:



Hình 1.9 : Các tiến trình hoạt động của Logstash

Logstash sử dụng kết hợp 2 cơ chế kéo (PULL) và đẩy (PUSH) để tổng hợp dữ liệu từ các nguồn dữ liệu đến đích lưu trữ:



Hình 1.10 : Cơ chế hoạt động Pull và Push của Logstash

#### a. Input Plugin

Input plugin làm nhiệm vụ thu thập dữ liệu log, mỗi khi có dữ liệu thay đổi trong tệp dữ liệu log, tiến trình thu thập trong Input Plugin sẽ được kích hoạt để triết xuất lấy phần dữ liệu đó.

Logstash cung cấp nhiều dạng input plugin tương ứng với nhiều nguồn dữ liệu khác nhau, input plugin làm nhiệm vụ kết xuất (extract) dữ liệu từ dữ liệu nguồn. Công việc này tương ứng với công đoạn Extract trong công nghệ tích hợp dữ liệu ETL.

Sau đây chúng ta sẽ tìm hiểu một số input plugin được sử dụng trong luận văn.

### ➤ File input

File input được sử dụng để đọc dữ liệu từ tệp (file), thường là file chứa dữ liệu log. Mặc định File input sẽ đọc dữ liệu trong tệp theo từng dòng, mỗi dòng được phân cách bởi ký tự “\n”. File có thể được cấu hình đọc ở 2 chế độ:

- Chế độ đọc đoạn cuối (Tail mode): Ở chế độ này, file được coi là nội dung không bao giờ kết thúc, file luôn được theo dõi những thay đổi về nội dung và sẽ chuyển những nội dung thay đổi đó sang đích. Khi file được quay vòng thì vị trí đọc hiện tại sẽ được đặt về 0 và tiếp tục đọc tiếp những thay đổi dữ liệu của file. Các file dữ liệu log thường được cấu hình ở chế độ này, và đây là chế độ đọc mặc định của file input.
- Chế độ đọc thông thường (Read mode): Ở chế độ này, file được coi như nội dung đã hoàn thành và không có phát sinh thêm dữ liệu. Khi đó File input sẽ thực hiện đọc từ đầu file đến cuối file và file chỉ được đọc 1 lần.

### ➤ JDBC input

JDBC input được sử dụng để kết nối tới bất kỳ loại cơ sở dữ liệu nào. Dữ liệu sẽ được lấy định kỳ từ cơ sở dữ liệu và đưa vào LogStash thông qua kết nối JDBC. Ví dụ cấu hình một JDBC input lấy dữ liệu định kỳ từ cơ sở dữ liệu MySQL:

```
input {
  jdbc {
    jdbc_driver_library => "mysql-connector-java-5.1.36-bin.jar"
    jdbc_driver_class => "com.mysql.jdbc.Driver"
    jdbc_connection_string => "jdbc:mysql://localhost:3306/mydb"
    jdbc_user => "mysql"
    parameters => { "favorite_artist" => "Beethoven" }
    schedule => "* * * * *"
    statement => "SELECT * from songs where artist = :favorite_artist"
  }
}
```

## **b. Filter Plugin**

Thực hiện nhiệm vụ của tiến trình Transform trong công nghệ ETL, đó là làm sạch, xử lý, chuyển đổi dữ liệu cần thiết để phù hợp với mục đích sử dụng.

Một số Filter Plugin được sử dụng trong luận văn:

### ➤ **Grok filter plugin**

Grok là một filter plugin quan trọng nhất của logstash để xử lý dữ liệu log. Grok được sử dụng để phân tích văn bản phi cấu trúc và cấu trúc hóa dữ liệu của văn bản đó. Do dữ liệu log của mỗi hệ thống là khác nhau và thường ở dạng văn bản phi cấu trúc nên bộ lọc “Grok” này rất hữu ích khi có thể được sử dụng để biến các dòng dữ liệu log phi cấu trúc thành một nguồn dữ liệu có cấu trúc để có thể sử dụng cho mục đích truy vấn và phân tích.

Bộ lọc Grok sử dụng biểu thức chính quy (Regular Expression) để kết xuất ra các dữ liệu phù hợp với biểu thức. Grok trong LogStash hỗ trợ một số biểu thức chính quy được chuẩn hóa như dạng DATE, MONTH, IP, ... Tuy nhiên nếu dữ liệu cần đọc không thể sử dụng được loại biểu thức chuẩn hóa có sẵn nào của LogStash thì ta có thể thực hiện viết cú pháp biểu thức chính quy thủ công.

### ➤ **Mutate filter plugin**

Với bộ lọc Mutate cho phép người dùng có thể thực hiện các biến đổi trên trường dữ liệu (field) của sự kiện được Logstash xử lý như: đổi tên, xóa, thay thế, ghi đè trường dữ liệu.

Bảng 1.4 : Một số ví dụ cấu hình Mutate plugin

Thuộc tính	Mô tả	Ví dụ
add_field	Thêm trường vào sự kiện được xử lý bởi Logstash	<pre>filter {   mutate {     <b>add_field</b> =&gt; { "foo_%{somefield}" =&gt; "Hello world, from %{host}" }   } }</pre>
convert	Chuyển đổi kiểu dữ liệu cho trường	<pre>filter {   mutate {     <b>convert</b> =&gt; { "fieldname" =&gt; "integer" }   } }</pre>
lowercase	Chuyển đổi dữ liệu kiểu chuỗi thành dạng chữ thường	<pre>filter {   mutate {     <b>lowercase</b> =&gt; [ "fieldname" ]   } }</pre>
merge	Ghép 2 trường dữ liệu kiểu chuỗi vào làm một	<pre>filter {   mutate {     <b>merge</b> =&gt; { "dest_field" =&gt; "added_field" }   } }</pre>

### ➤ Multiline filter plugin

Multiline input plugin được sử dụng để gộp nhiều dòng thông tin thành một sự kiện được xử lý bởi LogStash. Ví dụ cấu hình Multiline như sau:

```
multiline {
  pattern => "%{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{YEAR}"
  negate => true
  what => "previous"
}
```

Trong ví dụ trên, các dòng văn bản đằng sau đoạn văn bản thời gian thỏa mãn điều kiện của biểu thức chính quy “pattern” sẽ được gom lại thành một sự kiện để Logstash xử lý.

### c. Output Plugin

Output Plugin thực hiện nhiệm vụ Load trong công nghệ ETL, đó là đưa dữ liệu sau chuyển đổi từ Filter Plugin vào lưu trữ tại đích phục vụ các bài toán như lưu trữ, phân tích, giám sát, quản lý tập trung, ... LogStash cũng hỗ trợ rất nhiều kiểu Output Plugin tương ứng với nhiều kiểu lưu trữ dữ liệu như Elasticsearch, cơ sở dữ liệu quan hệ, cache, ...

Một số Output Plugin được sử dụng trong luận văn:

#### ➤ Elasticsearch Plugin

Mục đích sử dụng chính của Elasticsearch Plugin là phục vụ việc đưa dữ liệu đầu ra của filter plugin (dữ liệu sau chuyển đổi) vào đánh chỉ mục trong hệ truy hồi thông tin Elasticsearch phục vụ các bài toán về tìm kiếm, phân tích dữ liệu hoặc trừu tượng hóa dữ liệu trên các công cụ như Kibana.

Ngoài việc đánh chỉ mục cho dữ liệu được thu thập từ LogStash thì Elasticsearch plugin còn thực hiện được hầu hết các tác vụ khác như tạo, sửa, xóa chỉ mục trong Elasticsearch thông qua giao thức Http. Dữ liệu được truyền qua giao thức Http có thể được nén để tăng hiệu năng cũng như giảm băng thông mạng.



Bảng 1.5 : Một số cấu hình quan trọng đối với Output ElasticSearch plugin

Cấu hình	Mô tả
action	<p>Chỉ thị hành động, các giá trị có thể:</p> <ul style="list-style-type: none"> <li>- Index: đánh chỉ mục cho tài liệu</li> <li>- Delete: Xóa một tài liệu trong chỉ mục</li> <li>- Create: Tạo thêm tài liệu trong index của ElasticSearch.</li> <li>- Update: Bản chất là upsert, cập nhật tài liệu trong chỉ mục nếu tài liệu đã tồn tại, ngược lại thì sẽ tạo thêm tài liệu trong chỉ mục.</li> </ul>
http_compression	Cấu hình để cho phép có nén dữ liệu khi truyền tải qua giao thức Http hay không. Nếu muốn nén dữ liệu thì đặt tham số này bằng "true"
hosts	Chỉ định đường dẫn tới ElasticSearch Server để LogStash kết nối tới.

### ➤ Mail plugin

Mail plugin được sử dụng để gửi email thông báo khi có dữ liệu được đưa vào đầu ra (Output), ta có thể thiết lập điều kiện để kích hoạt việc gửi email hay không. Ví dụ cấu hình Mail plugin output:

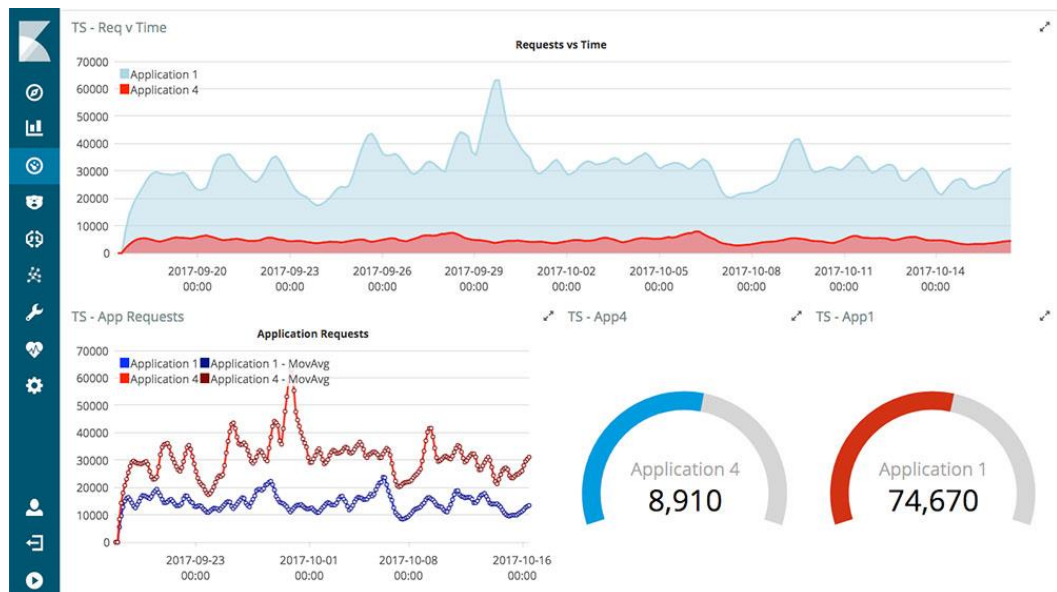
```
output {
  if "shouldmail" in [tags] {
    email {
      to => 'technical@example.com'
      from => 'monitor@example.com'
      subject => 'Alert - %{title}'
      body => "Tags: %{tags}\\\n\\Content:\\\n%{message}"
      template_file => "/tmp/email_template.mustache"
      domain => 'mail.example.com'
      port => 25
    }
  }
}
```

Ví dụ trên sẽ kích hoạt việc gửi email khi trong tài liệu có chứa từ “shouldmail”. Mail plugin rất hữu ích trong việc cảnh báo qua email khi tìm thấy mã lỗi trong dữ liệu log. Việc phát hiện và gửi email cảnh báo gần như là tức thì, giúp cho việc khắc phục kịp thời các lỗi để tránh xảy ra sự cố cho hệ thống.

#### 1.4.4. Kibana

Kibana là một phần mềm mã nguồn mở được sử dụng để trừu tượng hóa dữ liệu, xây dựng các biểu đồ, báo cáo, màn hình giám sát và phân tích dữ liệu thời gian thực từ nguồn dữ liệu trong Elasticsearch.

Kibana cung cấp giao diện cho phép người dùng có thể thực hiện truy vấn toàn văn trên hệ truy hồi thông tin Elasticsearch, xây dựng biểu đồ, các màn hình điều khiển từ dữ liệu chỉ mục trên Elasticsearch một cách nhanh chóng.



Hình 1.11 : Giới thiệu Kibana

#### 1.5. Kết luận

Trong Chương I, Luận văn đã trình bày tổng quan về bài toán cần giải quyết. Luận văn cũng đã trình bày về điểm mạnh, điểm yếu của 3 giải pháp công nghệ thông dụng cho bài toán quản lý dữ liệu log gồm: ELK, splunk, GrayLog và lựa chọn được giải pháp công nghệ sử dụng cho luận văn là giải pháp ELK vì đáp ứng được các tiêu chí như chi phí thấp, triển khai nhanh, tính mở rộng cao và sử dụng được cho phân tích dữ liệu lớn.

## CHƯƠNG II: PHÂN TÍCH, THIẾT KẾ, XÂY DỰNG HỆ THỐNG QUẢN LÝ LOG TẬP TRUNG CHO TẬP ĐOÀN BẢO VIỆT

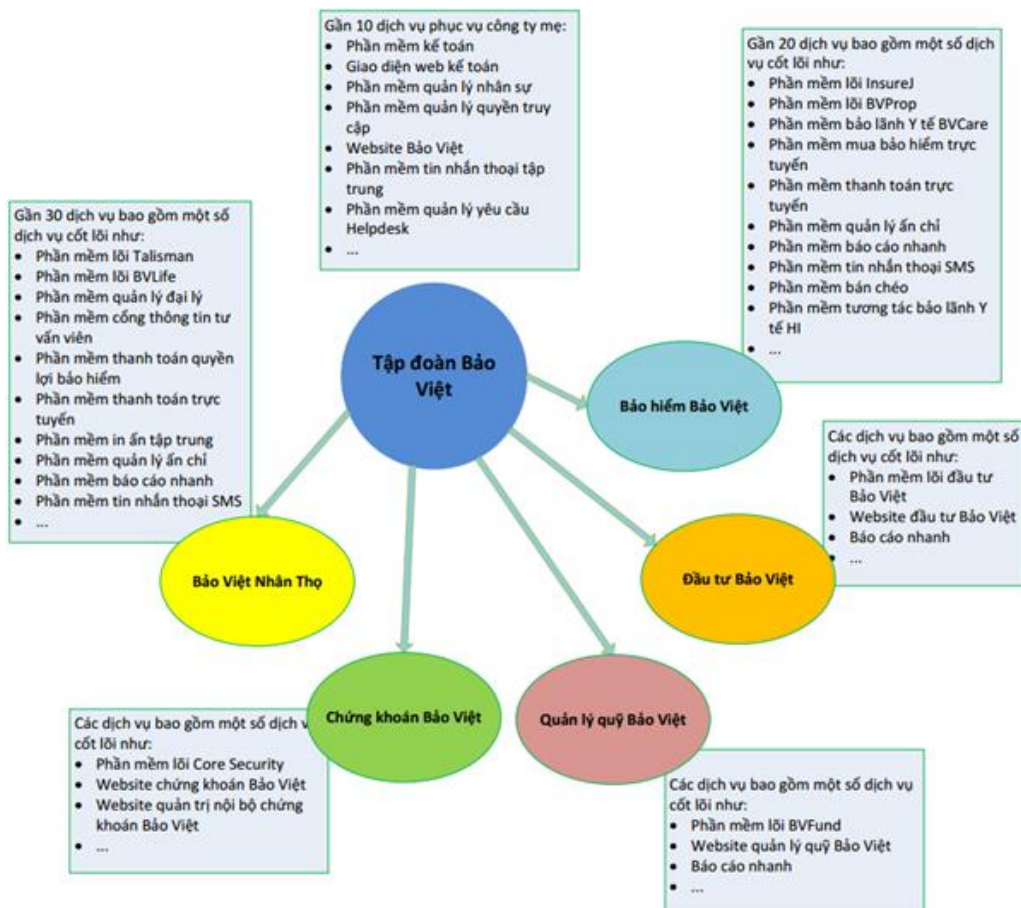
### 2.1. Hiện trạng hạ tầng CNTT Bảo Việt

#### 2.1.1. Hiện trạng dịch vụ

Hiện nay Tập đoàn Bảo Việt bao gồm các đơn vị thành viên như:

- Công ty mẹ Tập đoàn Bảo Việt
- Bảo hiểm Bảo Việt
- Bảo Việt nhân thọ
- Công ty quản lý quỹ Bảo Việt
- Công ty đầu tư Bảo Việt
- Công ty Chứng khoán Bảo Việt
- Ngân hàng Bảo Việt

Tập đoàn Bảo Việt quản lý và cung cấp các dịch vụ CNTT cho các đơn vị thành viên để phục vụ hoạt động sản xuất, kinh doanh của các đơn vị. Với đặc thù là Tập đoàn tài chính hoạt động trong tất cả các lĩnh vực tài chính, bảo hiểm, ngân hàng nên số lượng dịch vụ CNTT do Tập đoàn quản lý là rất lớn. Dưới đây là sơ đồ tổ chức và các dịch vụ cốt lõi mà Tập đoàn đang cung cấp cho các đơn vị thành viên:



Hình 2.1 : Cơ cấu tổ chức và mô hình dịch vụ tại Tập đoàn Bảo Việt

### 2.1.2. Hiện trạng hạ tầng máy chủ

Theo thống kê chưa đầy đủ, hệ thống máy chủ cả vật lý và ảo hóa cung cấp cho các dịch vụ công nghệ thông tin của Bảo Việt lên tới hàng nghìn máy.

*Bảng 2.1 : Số lượng máy chủ cung cấp cho các đơn vị thuộc Tập đoàn Bảo Việt*

<b>Đơn vị</b>	<b>Số lượng máy chủ (Vật lý + ảo hóa)</b>
Bảo Việt nhân thọ	~ 300 máy chủ
Bảo hiểm Bảo Việt	~ 200 máy chủ
Công ty mẹ Tập đoàn	~ 100 máy chủ
Công ty quản lý quỹ Bảo Việt	~ 10 máy chủ
Công ty đầu tư Bảo Việt	~ 10 máy chủ
Công ty chứng khoán Bảo Việt	~ 15 máy chủ

Bên cạnh các máy chủ còn có các thiết bị mạng, thiết bị lưu trữ đặc thù cũng cần phải được giám sát hoạt động thường xuyên.

### 2.1.3. Hiện trạng nền tảng hệ điều hành và phần mềm

Do dịch vụ cung cấp cho các công ty thành viên là rất nhiều và đa dạng nên nền tảng hệ điều hành và phần mềm sử dụng trong các dịch vụ công nghệ thông tin của Tập đoàn Bảo Việt cũng rất phong phú về chủng loại và phiên bản.

#### a. Hiện trạng nền tảng hệ điều hành

*Bảng 2.2 : Hiện trạng nền tảng hệ điều hành được sử dụng tại Tập đoàn Bảo Việt*

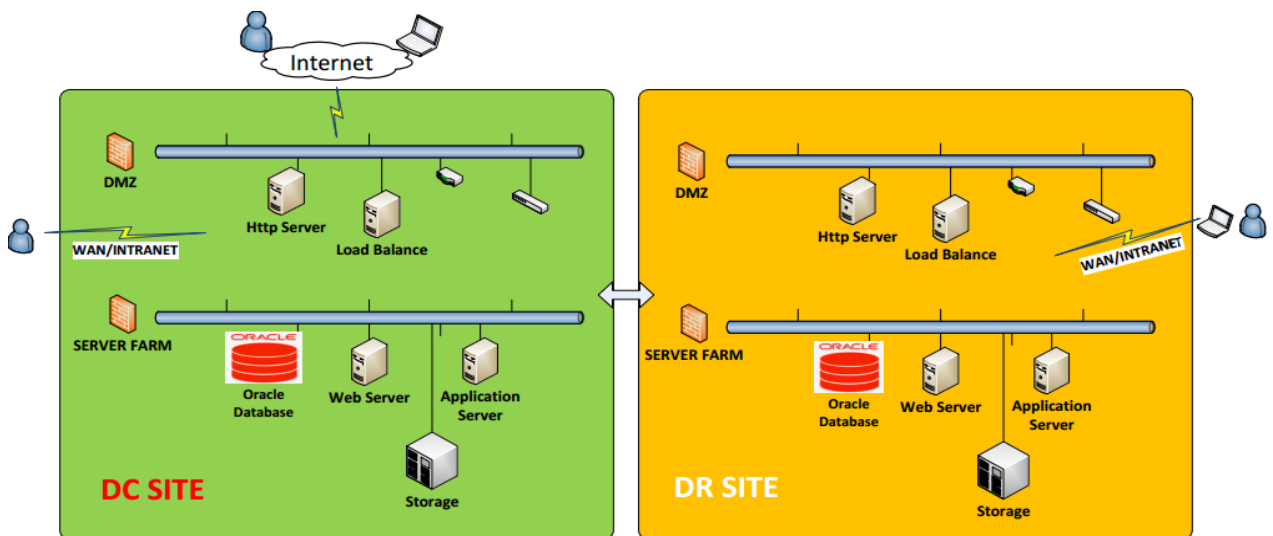
<b>Window</b>	<b>Linux</b>	<b>Oracle Solaris</b>	<b>HP-UX</b>
- Window Server 2003	- Linux RedHat 5 - Linux RedHat 6	- Solaris 10 - Solaris 11	- HP-UX 10 - HP-UX 11
- Window Server 2008	- Linux RedHat 7 - Oracle Linux 6		
- Window Server 2012	- Oracle Linux 7		
- Window Server 2016			

## b. Hiện trạng nền tảng phần mềm

Bảng 2.3 : Hiện trạng nền tảng phần mềm được sử dụng tại Tập đoàn Bảo Việt

Cơ sở dữ liệu	Phần mềm lớp giữa	Web Server	Http Server	Cân bằng tải
Oracle: 12c, 11g, 10g	Oracle Weblogic: 10g, 11g, 12c	Microsoft IIS	Apache Http Server	HA-Proxy
SQL Server: 2005, 2008	IBM WebSphere Application Server	Apache Tomcat	Oracle WebCache	Ngnix
MySQL, MariaDB	JBoss		Oracle Http Server	KeepAlive
PosgreSQL	Oracle Service Bus 12c		IBM Http Server	

### 2.1.4. Hiện trạng mô hình hạ tầng hệ thống CNTT



Hình 2.2 : Hiện trạng mô hình hạ tầng CNTT tại Bảo Việt

Hiện trạng hệ thống CNTT của Tập đoàn Bảo Việt bao gồm một hệ thống chính (Data Center -DC Site) cung cấp dịch vụ phục vụ trực tiếp cho người dùng, và một hệ thống dự phòng thảm họa để phòng ngừa sự cố, thảm họa trên hệ thống chính Data

Center. Hai hệ thống này được đặt tại 2 vị trí địa lý khác nhau, 2 hệ thống hạ tầng và mạng khác nhau.

Trong một hệ thống, được chia thành 2 lớp mạng là vùng DMZ và vùng Server Farm. Vùng DMZ được đặt các máy chủ chạy Http, Load Balance, F5, ... làm nhiệm vụ đón nhận những kết nối từ bên ngoài Internet vào hệ thống. Vùng Server Farm đặt các máy chủ chạy các ứng dụng Web, Application, Cơ sở dữ liệu, ... Giữa các vùng mạng đều được đặt Firewall để đảm bảo an ninh, an toàn cho hệ thống.

### **2.1.5. Hiện trạng quản lý, giám sát hệ thống**

Hiện tại Bảo Việt đang giám sát hệ thống một cách khá thủ công. Hàng ngày nhóm trực hệ thống sẽ thực hiện truy xuất định kỳ (từ 1 đến 3 lần tùy theo hệ thống) vào nơi lưu dữ liệu log trên các máy chủ và thực hiện đọc, tìm các mã lỗi một cách thủ công theo hướng dẫn của cán bộ quản trị trên các tệp dữ liệu log đó. Nếu phát hiện được mã lỗi, cán bộ trực hệ thống sẽ gửi thư điện tử thông báo tới cán bộ quản trị được biết để thực hiện kiểm tra và khắc phục lỗi.

Chúng tôi nhận thấy với cách thức kiểm tra dữ liệu log như hiện nay tồn tại các hạn chế sau:

- Cán bộ phải truy xuất dữ liệu thủ công và phân tán trên các máy chủ để đọc và tìm mã lỗi.
- Việc tìm kiếm thủ công sẽ rất chậm và có thể gây ra nhầm lẫn hoặc thiếu sót trong quá trình kiểm tra dữ liệu log, gây rủi ro sẽ xảy ra sự cố cho hệ thống mà người quản trị không được biết kịp thời.
- Dữ liệu log không được tận dụng để xây dựng các báo cáo và phân tích để tìm ra những thông tin hữu ích.
- Mất nhiều nguồn lực con người để thực hiện các công việc đọc và tìm kiếm lỗi thủ công trong khi có thể tự động hóa được công việc này, giải phóng được nhân lực.

### **2.2. Kiến trúc giải pháp**

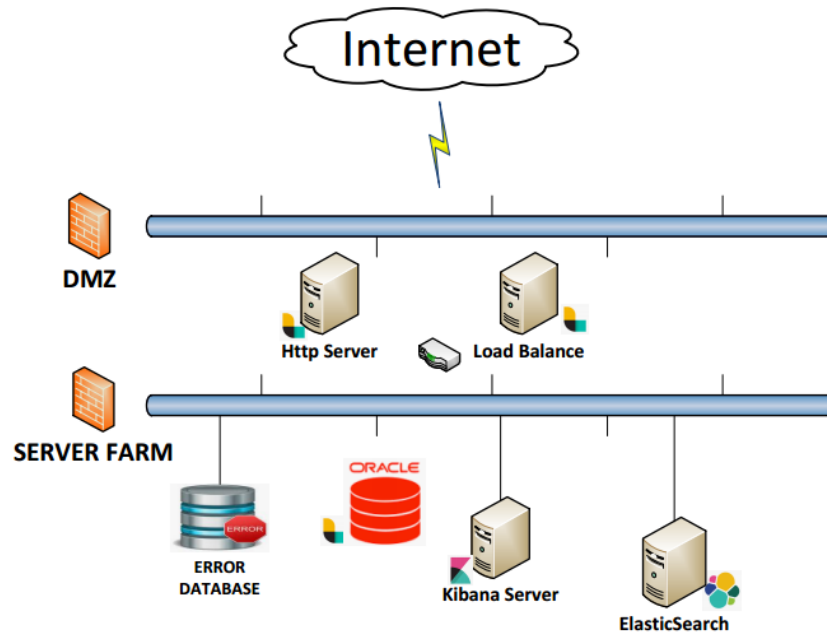
Như đã phân tích, hai vấn đề lớn cần giải quyết tại Tập đoàn Bảo Việt cho bài toán quản lý dữ liệu log là:

- Cần quản lý dữ liệu log tập trung phục vụ bài toán tìm kiếm, phân tích một cách nhanh và hiệu quả.
- Cần phát hiện và cảnh báo tự động các lỗi trong dữ liệu log để kịp thời khắc phục trước khi sự cố xảy ra với hệ thống, giảm thiểu các công việc thủ công thiếu chính xác trong việc giám sát và phân tích dữ liệu log.

Từ đó, mô hình giải pháp sử dụng cần phải đáp ứng và giải quyết được các yêu cầu trên.

### 2.2.1. Mô hình tổng thể giải pháp

Từ hiện trạng hạ tầng CNTT của Tập đoàn Bảo Việt đã phân tích ở mục 2.1, mô hình tổng thể giải pháp quản lý dữ liệu log tập trung được thiết kế như sau:



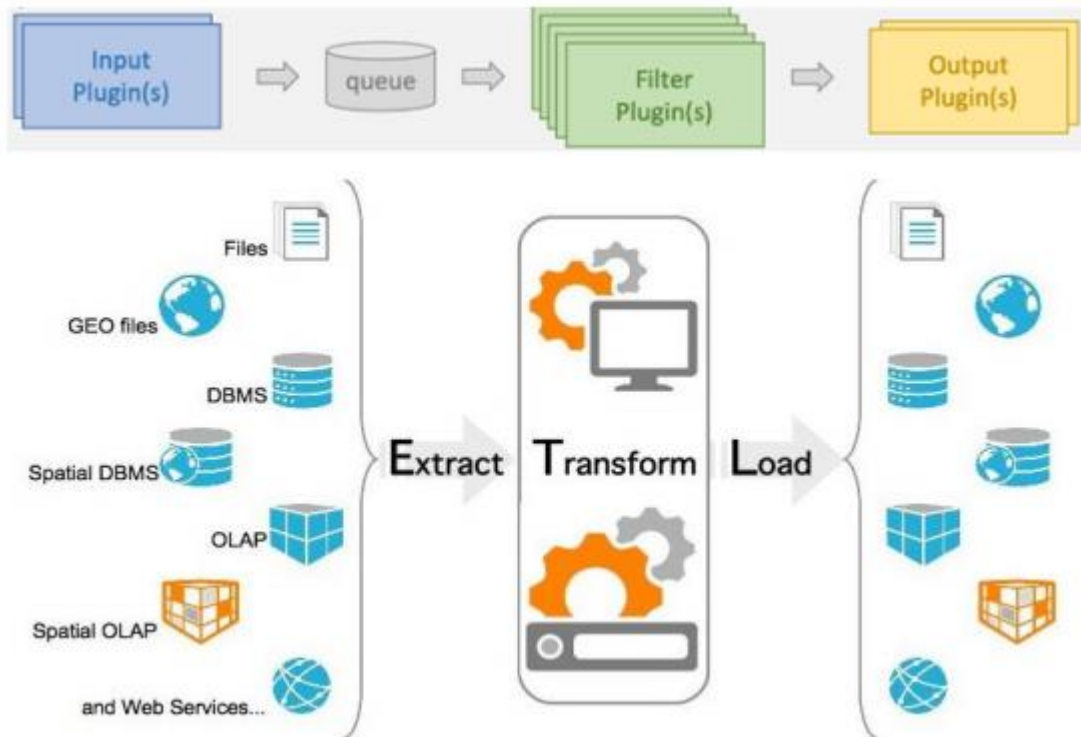
Hình 2.3 : Mô hình tổng thể giải pháp

Trong đó:

+ Xây dựng một cụm ElasticSearch Cluster đặt trong vùng “Server Farm” làm một hệ truy hồi thông tin, thực hiện nhiệm vụ đánh chỉ mục cho dữ liệu được thu thập từ Logstash phục vụ bài toán tìm kiếm và phân tích.

+ Một máy chủ Kibana Server đặt trong vùng “Server Farm” làm nhiệm vụ xây dựng giao diện cho phép người dùng trừu tượng hóa dữ liệu, xây dựng các báo cáo, bảng điều khiển để theo dõi tình trạng của hệ thống, các lỗi mà hệ thống đang gặp phải có thể gây nên sự cố cho hoạt động của hệ thống.

+ Logstash được cài đặt lên các máy chủ cần thu thập dữ liệu log. Dữ liệu được Logstash sử dụng công nghệ ETL xử lý qua ba giai đoạn:



Hình 2.4 : Mô hình hoạt động của Logstash

Trong đó:

- **Input:** Thu thập dữ liệu từ nguồn, bước này tương ứng với công đoạn Extract trong công nghệ ETL.
- **Filter:** Chuyển đổi, lọc, làm sạch dữ liệu từ bước thu thập dữ liệu, bước này tương ứng với công đoạn Transform trong công nghệ ETL.
- **Output:** Bước này tương ứng với công đoạn Load trong công nghệ ETL, đưa dữ liệu sau thu thập và làm sạch qua 2 bước input và filter vào lưu trữ tại đích, đích có thể là cơ sở dữ liệu, file, hệ truy hồi thông tin, ... để phục vụ truy vấn, tìm kiếm hoặc phân tích.

+ Xây dựng một cơ sở dữ liệu chứa các mã lỗi có thể có đối với từng loại dữ liệu log của từng loại hệ thống, phần mềm. Cơ sở dữ liệu này sẽ được Logstash đọc để làm danh mục các mã lỗi mỗi khi Logstash thực hiện tiến trình “Filter”. Với cơ sở dữ liệu chứa mã lỗi này, nếu có phát sinh thêm mã lỗi cần kiểm soát ta chỉ cần thêm mã lỗi vào cơ sở dữ liệu. Cơ sở dữ liệu chứa mã lỗi này được sử dụng trên nền tảng cơ sở dữ liệu mã nguồn mở mariaDB. Cơ sở dữ liệu được thiết kế với 3 bảng chính:



Tên bảng	Mục đích
DM_PHANMEM	Bảng này chứa danh mục các phần mềm được sử dụng trong các hệ thống.
DM_MAYCHU	Bảng này chứa danh mục các máy chủ đang được sử dụng trong các hệ thống, và cho biết máy chủ thuộc dịch vụ nào của Bảo Việt đang cung cấp và đang chạy phần mềm nào trên máy chủ đó.
DM_MALOI	Bảng map giữa phần mềm và mã lỗi, cho biết mã lỗi có thể có của các phần mềm. Khi muốn thêm mã lỗi mới của một phần mềm nào đó thì thêm bản ghi vào bảng này.

Cấu trúc của các bảng được thiết kế như sau:

- DM\_MAYCHU:

Tên cột	Ý nghĩa
ID	ID tự tăng của bản ghi
IP	Địa chỉ IP của máy chủ
LOG_PATH	Nơi lưu trữ dữ liệu log đang đọc trên máy chủ
DICHVU	Tên dịch vụ mà máy chủ đang được sử dụng
ID_PHANMEM	Phần mềm đang được sử dụng ứng với IP và đường dẫn nơi lưu dữ liệu log

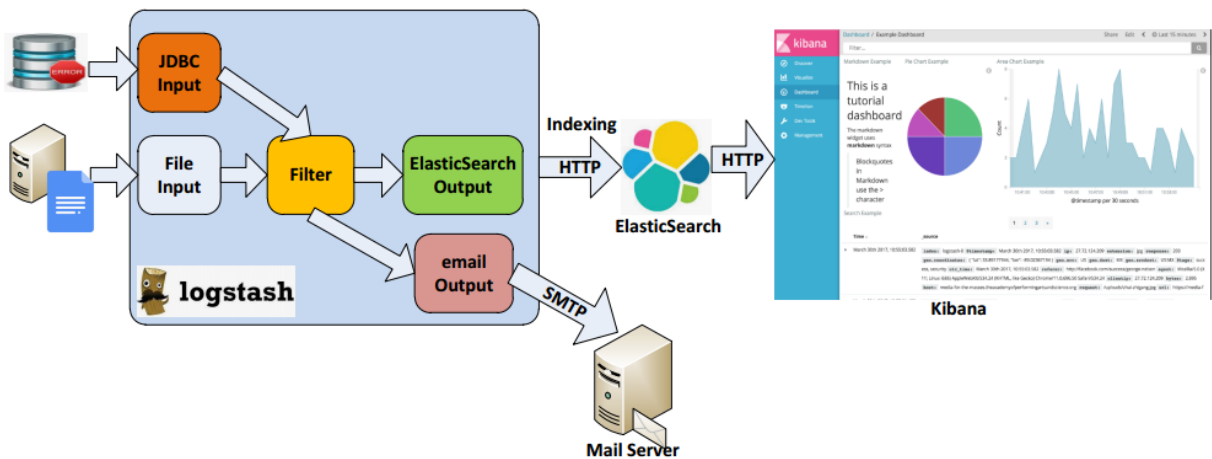
- DM\_PHANMEM:

Tên cột	Ý nghĩa
ID	Mã tự tăng
MAPM	Mã phần mềm
LOAIPM	Loại phần mềm (Database hay Web hay Http, ...)

- DM\_MALOI:

Tên cột	Ý nghĩa
ID	Mã tự tăng của bản ghi
MALOI	Mã lỗi
ID_PHANMEM	ID của phần mềm trong bảng DM_PHANMEM

## 2.2.2. Mô hình luồng dữ liệu



Hình 2.5 : Mô hình luồng dữ liệu giải pháp

Theo như mô hình trên, luồng dữ liệu sẽ xuất phát từ tệp dữ liệu log và đi theo luồng sau:

+ Tệp dữ liệu log sẽ được đọc nội dung bởi “File Input Plugin” của Logstash theo tần xuất nhất định (có thể cấu hình được tần xuất đọc tệp).

+ Bên cạnh đó còn có JDBC Input plugin làm nhiệm vụ đọc mã lỗi trong cơ sở dữ liệu mã lỗi để phục vụ công đoạn Filter tìm mã lỗi trong dữ liệu log đọc được từ File Input plugin.

+ Dữ liệu được đọc bởi “File Input” sẽ được chuyển tiếp đến “Filter Plugin”. Ở đây dữ liệu sẽ được lọc, làm sạch, biến đổi theo đặc thù của từng loại dữ liệu log. Tìm mã lỗi trong nội dung log. Đầu ra trong bước Filter này là tài liệu dạng JSON chứa nội dung thông điệp của log và mã lỗi nếu được tìm thấy.

+ Tài liệu JSON từ bước “Filter” được đưa tới các Output plugin của Logstash. Ở đây ta thiết kế sử dụng 2 Output Plugin là “email Output” và “ElasticSearch Output” để giải quyết 2 bài toán khác nhau.

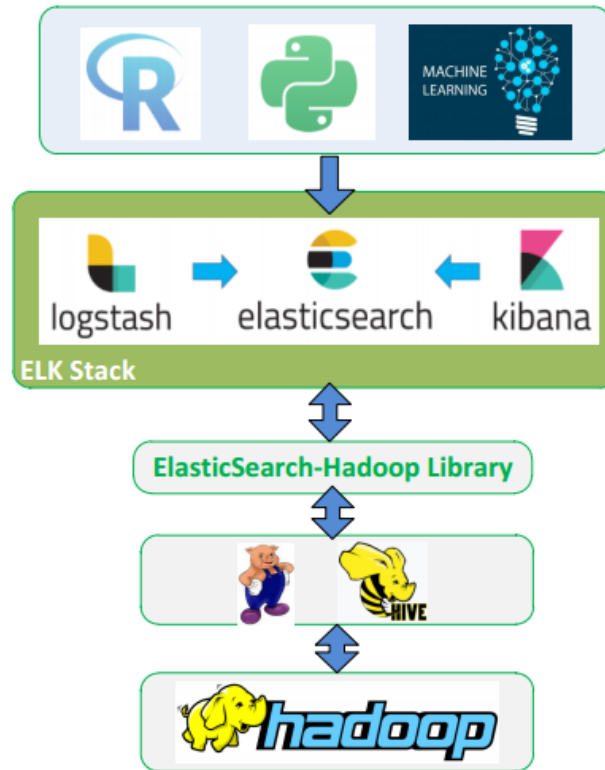
+ Dữ liệu JSON được đưa tới “email Output” để gửi thư điện tử tự động đến cán bộ quản trị hệ thống khi có mã lỗi được tìm thấy trong dữ liệu log.

+ Dữ liệu JSON được đưa tới “ElasticSearch Output” để thực hiện đánh chỉ mục cho tài liệu JSON đó trong hệ truy hồi thông tin ElasticSearch phục vụ bài toán tìm kiếm, trực quan hóa dữ liệu, xây dựng báo cáo và phân tích dữ liệu log.

+ Dữ liệu sau khi được đánh chỉ mục trong ElasticSearch sẽ được trực quan hóa, xây dựng báo cáo, xây dựng các màn hình giám sát, điều khiển trên Kibana.

### 2.2.3. Mô hình trao đổi dữ liệu với các hệ thống khác

Dữ liệu sau khi được đánh chỉ mục trên ElasticSearch có thể được sử dụng bởi các hệ thống khác với mục đích tìm kiếm toàn văn, làm nguồn dữ liệu để phân tích với các ngôn ngữ phân tích dữ liệu phổ biến (R, Python). Dữ liệu log được lưu trong ElasticSearch cũng sẽ là một nguồn dữ liệu đầu vào cho hệ thống dữ liệu lớn (BigData) phục vụ các bài toán phân tích dữ liệu lớn.



Hình 2.6 : Mô hình trao đổi dữ liệu với các hệ thống khác

### 2.3. Kết luận

Trong Chương II, Luận văn đã trình bày về hiện trạng hạ tầng dịch vụ, máy chủ và phần mềm đang được sử dụng tại Tập đoàn Bảo Việt, đây là thông tin đầu vào để chúng tôi có thể phân tích, thiết kế được kiến trúc của giải pháp.

Luận văn cũng đã trình bày về mô hình triển khai giải pháp quản lý log tập trung tại Tập đoàn Bảo Việt, áp dụng công nghệ ELK.

## CHƯƠNG III: XÂY DỰNG THỬ NGHIỆM HỆ THỐNG QUẢN LÝ LOG TẠI BẢO VIỆT

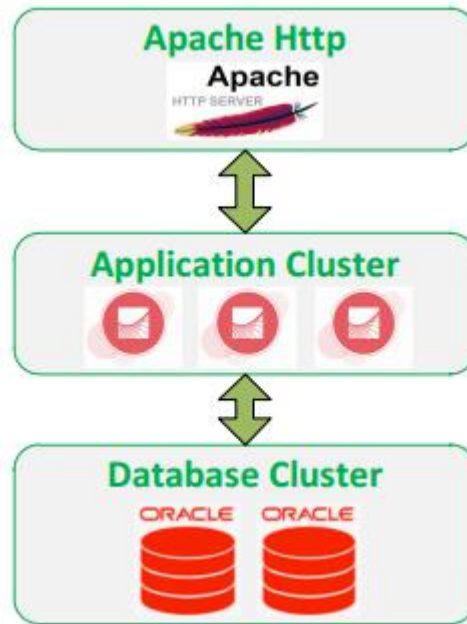
### 3.1. Môi trường thử nghiệm

Do hệ thống CNTT của tập đoàn Bảo Việt là rất lớn, đa dạng hóa về phần mềm, nền tảng, nên công đoạn triển khai sẽ được thực hiện theo từng giao đoạn. Mỗi giai đoạn sẽ thực hiện cấu hình cho một đơn vị mà Tập đoàn cung cấp dịch vụ CNTT (như Bảo hiểm nhân thọ, Bảo hiểm phi nhân thọ, Chứng khoán, ...) và được cuốn chiếu theo từng đầu dịch vụ. Trong phạm vi của luận văn, chúng tôi thực hiện thử nghiệm xây dựng hệ thống quản lý dữ liệu log cho hệ thống dịch vụ BVCare cung cấp cho đơn vị thành viên Bảo Hiểm phi nhân thọ, các máy chủ của dịch vụ đặt tại trung tâm dữ liệu (Data Center Site). Hệ thống BVCare được lựa chọn để triển khai đầu tiên do BVCare là một trong những hệ thống lớn của Tập đoàn Bảo Việt bao gồm gần như đầy đủ các thành phần máy chủ Http, máy chủ ứng dụng (Oracle Application Server) và cơ sở dữ liệu (Oracle database), đặc biệt BVCare là hệ thống cần phải hoạt động 24/7 để phục vụ hoạt động sản xuất kinh doanh của các công ty thành viên.

Dựa trên dữ liệu log, chúng tôi xây dựng màn hình giám sát, tìm kiếm, phát hiện lỗi tự động thay thế quá trình xử lý dữ liệu log thủ công trước đây. Quá trình thử nghiệm được thực hiện qua các công đoạn sau:

- Cài đặt Elasticsearch Cluster
- Cài đặt Logstash lên máy chủ chứa dữ liệu log cần giám sát
- Cài đặt máy chủ Kibana
- Xây dựng cơ sở dữ liệu chứa mã lỗi của cơ sở dữ liệu Oracle
- Cấu hình 3 thành phần của Logstash (input, filter và output) để trích xuất, tổng hợp dữ liệu log và đưa vào Elasticsearch để đánh chỉ mục
- Cấu hình Logstash để thực hiện tìm mã lỗi và gửi email thông báo tự động đến quản trị viên khi mã lỗi được tìm thấy trong dữ liệu log
- Xây dựng các màn hình điều khiển, giám sát tất cả các biến đổi xảy ra trong dữ liệu log một cách tức thời
- Sử dụng Kibana làm công cụ trừ tượng hóa dữ liệu để phân tích dữ liệu log

+ Mô hình hệ thống dịch vụ BVCare:



Hình 3.1: Mô hình hệ thống dịch vụ BVCare

Sau đây là bảng khảo sát các phân hệ trong dịch vụ BVCare và lượng dữ liệu log cần xử lý hàng ngày:

Bảng 3.1 : Bảng khảo sát dịch vụ BVCare

Site	Phân hệ	Số lượng máy chủ	Dung lượng log / ngày (MB)
Trung tâm dữ liệu	Cơ sở dữ liệu Oracle	2	~100
	Ứng dụng	4	~80
	Web	2	~50
Trung tâm dự phòng thảm họa	Cơ sở dữ liệu Oracle	2	~20
	Ứng dụng	2	~10
	Web	1	~5
Tổng		13	265

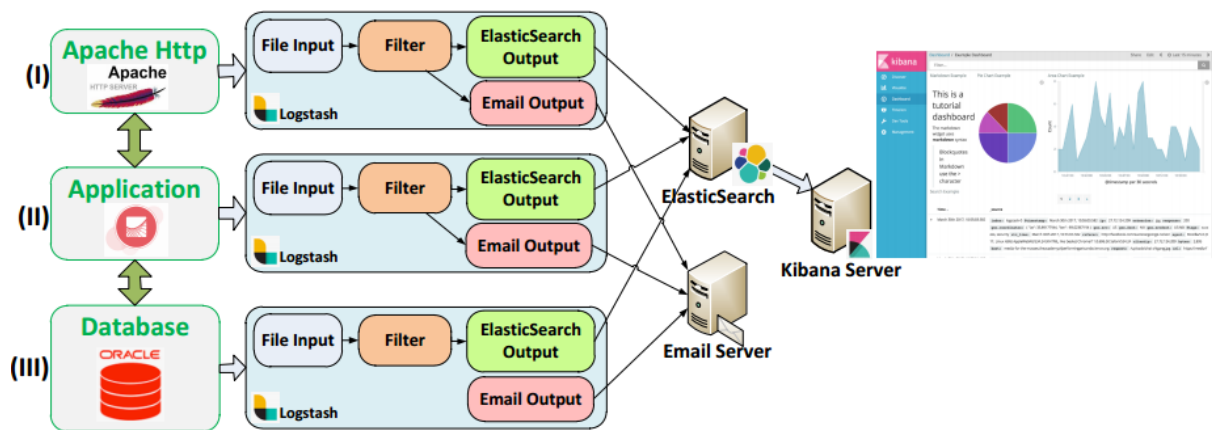
Qua khảo sát, trong phạm vi thử nghiệm đối với hệ thống dịch vụ BVCare, có 3 loại dữ liệu log cần thu thập gồm: Cơ sở dữ liệu Oracle, Ứng dụng Oracle Weblogic và Apache Http. Dung lượng log sinh ra trong ngày đối với 3 loại dữ liệu log này của hệ thống BVCare vào khoảng 265 MB. Qua nhận định của chúng tôi, dung lượng log sinh

ra trong ngày không phải quá lớn, Elasticsearch với kiến trúc Cluster có thể mở rộng theo chiều ngang (thêm các node) giúp tăng sức mạnh tính toán phân tán, hoàn toàn có thể xử lý được lượng dữ liệu log sinh ra bởi các hệ thống phần mềm của Tập đoàn Bảo Việt.

Dịch vụ BVCare là phần mềm quản lý bảo hiểm và bảo lãnh Y tế, được cung cấp cho Tổng công ty Bảo hiểm Bảo Việt, các công ty con và các cơ sở Y tế là đối tác với Bảo hiểm Bảo Việt sử dụng 24/7. Với đặc thù của dịch vụ cần hoạt động liên tục không có thời gian dừng, nên công việc giám sát và phát hiện kịp thời các lỗi gặp phải với hệ thống một cách nhanh nhất và chính xác nhất giúp giảm thiểu sự cố xảy ra với hệ thống.

### 3.2. Mô hình và cấu hình thử nghiệm

Từ mô hình hệ thống được trình bày trong mục 3.1 ta có mô hình cấu hình thử nghiệm khi triển khai giải pháp ELK như sau:



Hình 3.2 : Mô hình cấu hình thử nghiệm giải pháp ELK cho dịch vụ BVCare

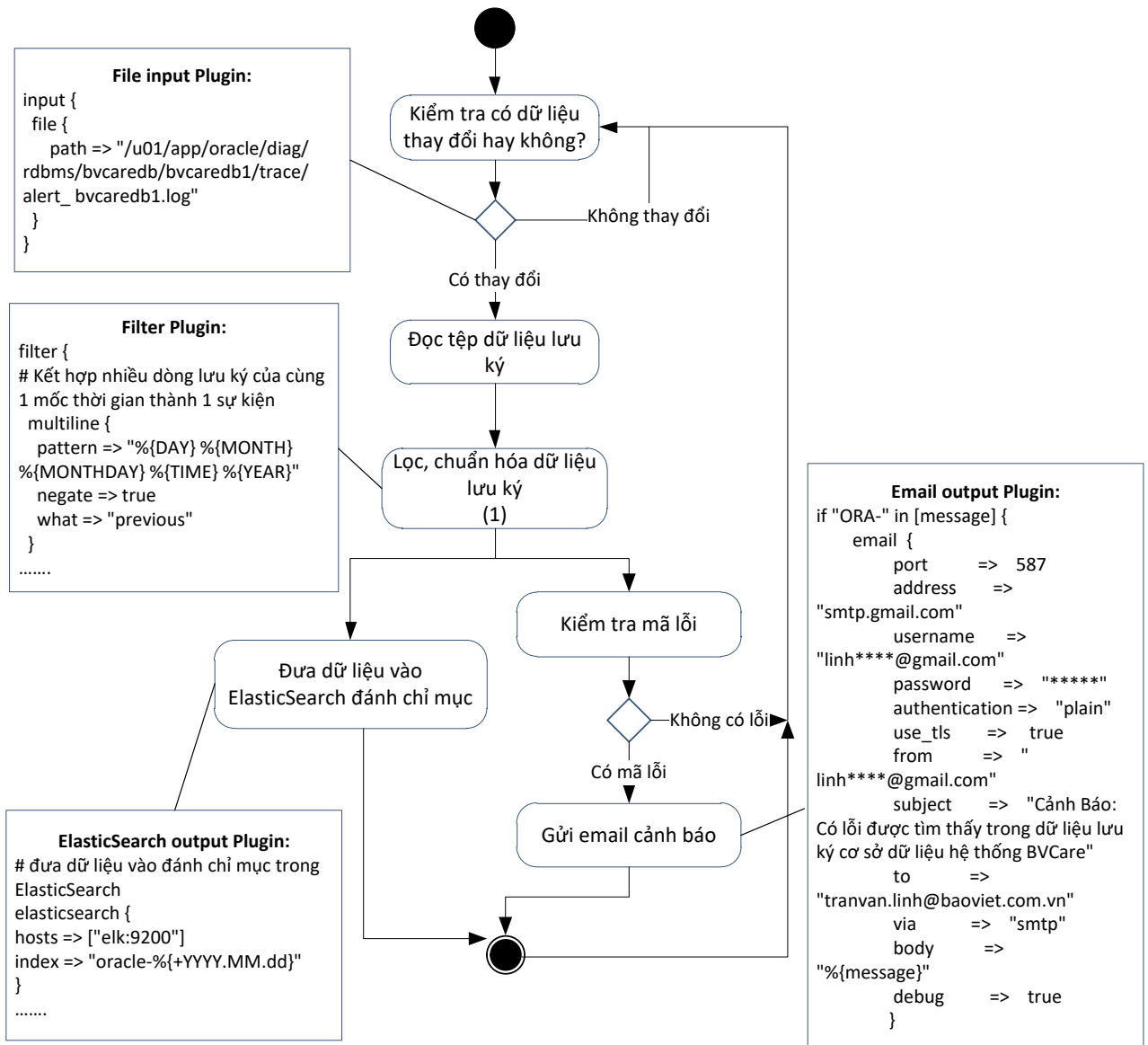
Mô hình thử nghiệm đối với hệ thống BVCare trong hình 3.2 bao gồm 3 luồng dữ liệu:

- (I) : Luồng trích xuất, tích hợp, giám sát, cảnh báo lỗi dữ liệu log trên máy chủ Apache Http.
- (II) : Luồng trích xuất, tích hợp, giám sát, cảnh báo lỗi dữ liệu log trên các máy chủ ứng dụng.
- (III) : Luồng trích xuất, tích hợp, giám sát, cảnh báo lỗi dữ liệu log trên các máy chủ cơ sở dữ liệu.

Dữ liệu log được Logstash trích xuất và tổng hợp theo công nghệ ETL, dữ liệu từ Logstash được đưa vào 2 đầu ra là Elasticsearch để đánh chỉ mục và Email Server để gửi email cảnh báo tự động nếu lỗi được phát hiện trong dữ liệu log. Sau đó dữ liệu

sau khi được đánh chỉ mục trong Elasticsearch được sử dụng bởi Kibana để người dùng, các nhà phân tích dữ liệu xây dựng các biểu đồ, trừu tượng hóa dữ liệu để phân tích chúng.

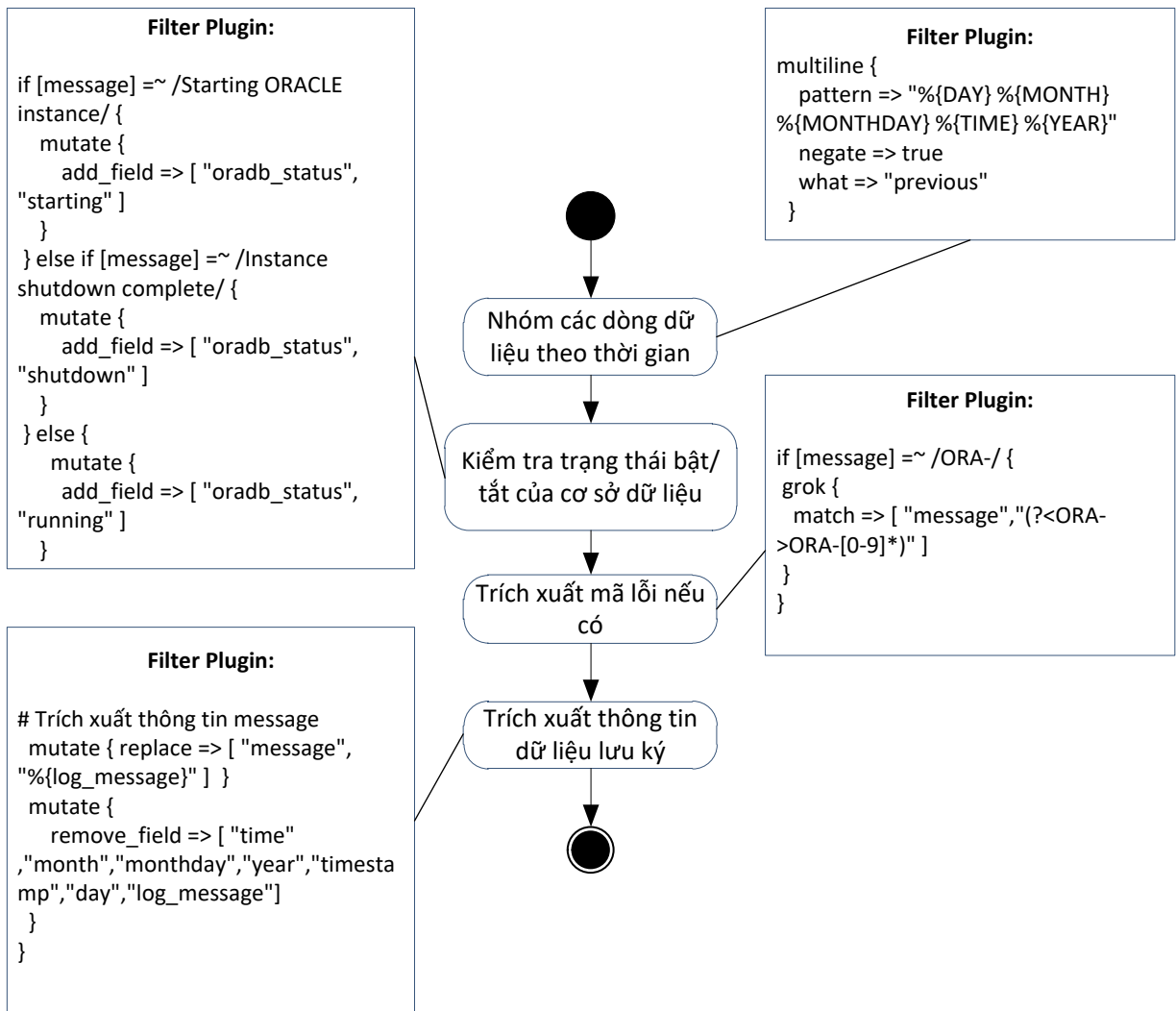
Với mỗi thành phần trong hệ thống (Http, ứng dụng, cơ sở dữ liệu) sẽ có dữ liệu log với các cấu trúc khác nhau, tuy nhiên để cấu hình Logstash trích xuất và tích hợp dữ liệu log vào Elasticsearch và gửi email cảnh báo tự động khi phát hiện mã lỗi ta phải thực hiện cấu hình cho 3 thành phần: Input, Filter và Output tương ứng với 3 tiến trình trong công nghệ ETL là Extract, Transform và Load. Chúng tôi xây dựng sơ đồ khối chung để xử lý cho nhiều loại dữ liệu log như sau:



Hình 3.3 : Sơ đồ khối luồng xử lý dữ liệu log với Logstash

Trong đó:

Bước (1): “Loại và chuẩn hóa dữ liệu log” được thực hiện theo lưu đồ sau:



Hình 3.4 : Sơ đồ khối bước “Lọc và chuẩn hóa dữ liệu log” thực hiện trong Logstash

Để cụ thể hơn việc áp dụng sơ đồ khối trên vào việc trích xuất, tích hợp và cảnh báo lỗi tự động cho dữ liệu log như thế nào, tiếp theo chúng tôi sẽ trình bày chi tiết cấu hình cho phân hệ cơ sở dữ liệu của hệ thống BVCare. Dữ liệu log thử nghiệm của cơ sở dữ liệu BVCare có dạng như sau:



```

.....
Mon Jan 28 15:42:41 2019
db_recovery_file_dest_size of 3882 MB is 0.00% used. This is a
user-specified limit on the amount of space that will be used by this
database for recovery-related files, and does not reflect the amount of
space available in the underlying filesystem or ASM diskgroup.
Mon Jan 28 15:49:37 2019
alter database open
Errors in file /u01/app/oracle/diag/rdbms/dr1/DR1/trace/DR1_ora_11881.trc:
ORA-01113: file 1 needs media recovery
ORA-01110: data file 1: '/u02/oradata/DR1/system01.dbf'
ORA-1113 signalled during: alter database open...
Mon Jan 28 15:49:38 2019
Checker run found 5 new persistent data failures
.....

```

Chi tiết cấu hình cho 3 thành phần input, filter và output của Logstash để trích xuất, tích hợp dữ liệu log và đưa vào Elasticsearch, gửi email cảnh báo tự động khi mã lỗi “ORA-“ được phát hiện trong dữ liệu log như sau:

```

# Cấu hình File input Plugin

input {
  file {
    path => "/u01/app/oracle/diag/rdbms/bvcaedb/bvcaedb1/trace/alert_
bvcaedb1.log"
  }
}

# Cấu hình Filter Plugin

filter {

# Kết hợp nhiều dòng log của cùng 1 mốc thời gian thành 1 sự kiện

```

```

multiline {
    pattern => "%{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{YEAR}"
    negate => true
    what => "previous"
}

# Tạo thêm trường để mô tả trạng thái của database:
# oradb_status: starting, running, shutdown
if [message] =~ /Starting ORACLE instance/ {
    mutate {
        add_field => [ "bvcaedb_status", "starting" ]
    }
} else if [message] =~ /Instance shutdown complete/ {
    mutate {
        add_field => [ "bvcaedb_status", "shutdown" ]
    }
} else {
    mutate {
        add_field => [ "bvcaedb_status", "running" ]
    }
}

# Tìm kiếm mã lỗi ORA- và tạo trường lưu mã lỗi tìm thấy
if [message] =~ /ORA-/ {
    grok {
        match => [ "message", "(?<ORA->ORA-[0-9]*)" ]
    }
}

# Trích xuất dữ liệu về thời gian và chi tiết của thông điệp

```

```

grok {
  match => [ "message", "%{DAY:day} %{MONTH:month}
%{MONTHDAY:monthday} %{TIME:time}
%{YEAR:year}(?<log_message>.*$)" ]
}

mutate {
  add_field => {
    "timestamp" => "%{year} %{month} %{monthday} %{time}"
  }
}

date {
  locale => "en"
  match => [ "timestamp" , "yyyy MMM dd HH:mm:ss" ]
}

# Trích xuất thông tin thông điệp

mutate { replace => [ "message", "%{log_message}" ] }

mutate {
  remove_field => [ "time"
, "month", "monthday", "year", "timestamp", "day", "log_message" ]
}

}

# Cấu hình Elasticsearch Output Plugin

output {
# đưa dữ liệu vào đánh chỉ mục trong Elasticsearch

elasticsearch {
hosts => ["elk:9200"]
index => "bvcare"
}
}

```

```
# gửi email cảnh báo khi mã lỗi "ORA-" được tìm thấy
if "ORA-" in [message] {
    email {
        port      => 587
        address   => "smtp.gmail.com"
        username  => "linh****@gmail.com"
        password  => "*****"
        authentication => "plain"
        use_tls   => true
        from      => "linh****@gmail.com"
        subject   => "Cảnh Báo: Có lỗi được tìm thấy trong dữ liệu log cơ
sở dữ liệu hệ thống BVCare"
        to       => "tranvan.linh@baoviet.com.vn"
        via      => "smtp"
        body     => "%{message}"
        debug    => true
    }
}
}
```

### 3.3. Kết quả đạt được

Sau khi triển khai thử nghiệm giải pháp, chúng tôi có thể nhận biết được lỗi phát sinh trong hệ thống theo thời gian thực, so với việc giám sát dữ liệu log thủ công trước đây thì kết quả này là một bước tiến giúp cán bộ quản lý hệ thống có thể khắc phục kịp thời các lỗi trước khi sự cố xảy ra với hệ thống, nâng cao rất tốt chất lượng dịch vụ. Với thông tin log được đánh chỉ mục trong Elasticsearch chúng tôi có thể giám sát được tình trạng hoạt động của hệ thống thông qua các màn hình điều khiển mà chúng tôi xây dựng trên Kibana, ngoài ra chúng tôi cũng có thể sử dụng dữ liệu log này để phân tích hành vi của người dùng và vấn đề an ninh, bảo mật đối với hệ thống của chúng tôi.

Để đánh giá được hiệu quả của hệ thống quản lý log chúng tôi xây dựng tình huống giả định rằng phân vùng lưu trữ dữ liệu của cơ sở dữ liệu sắp hết do phát sinh dữ liệu nhiều đột ngột, mà nếu không được bổ sung thêm dung lượng lưu trữ kịp thời

thì hệ thống sẽ bị ngừng hoạt động, khi đó trong dữ liệu log của cơ sở dữ liệu sẽ có cảnh báo “warning” và mã lỗi kèm theo.

Theo cách thức giám sát và kiểm tra hệ thống thủ công như hiện tại, cán bộ trực hạ tầng sẽ chỉ kiểm tra dữ liệu log định kỳ 1 ngày từ 1 đến 3 lần tùy theo hệ thống thì sẽ rất khó để phát hiện và xử lý kịp thời lỗi ở trên do lỗi có thể xảy ra tại bất kỳ thời điểm nào trong ngày, chưa kể đến việc tìm kiếm mã lỗi một cách thủ công có thể gây thiếu sót và bỏ qua lỗi, khi đó sự cố chắc chắn sẽ xảy ra đối với hệ thống, và việc khắc phục là quá muộn, sẽ ảnh hưởng đến hoạt động kinh doanh.

Tuy nhiên, khi đưa giải pháp ELK vào xây dựng hệ thống quản lý log, dữ liệu log được tổng hợp theo thời gian thực (real-time). Ngay sau khi trong dữ liệu log xuất hiện mã lỗi liên quan đến dung lượng lưu trữ bị hết, lập tức hệ thống sẽ tự động gửi email cảnh báo đến các cán bộ quản trị hệ thống để có thể thực hiện bổ sung thêm dung lượng ngay lập tức, trước khi sự cố xảy ra:



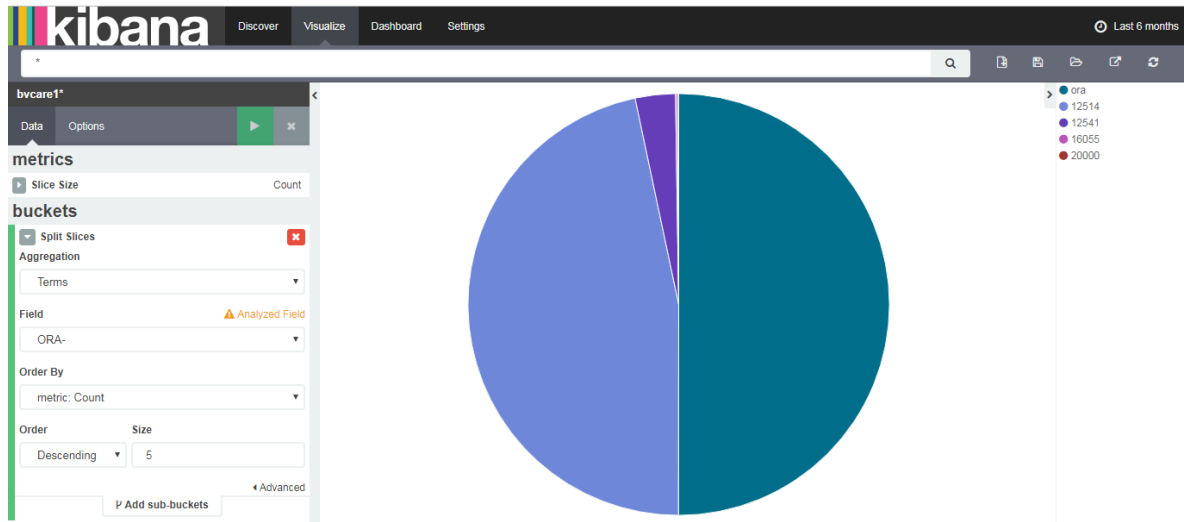
**Cảnh Báo:** Có lỗi được tìm thấy trong dữ liệu lưu ký cơ sở dữ liệu hệ thống BVCare  
**Linh Tran** đến: tranvan.linh

\*\*\* 2019-03-05 :10:04.164 4515 kcr.c

ORA-19815: WARNING: db\_recovery\_file\_dest\_size of 53687091200 bytes is 100.00% used, and has 2340864 remaining bytes available.

*Hình 3.5 : Email cảnh báo hết dung lượng lưu trữ*

Việc đưa dữ liệu log vào lưu trữ và đánh chỉ mục trong Elasticsearch còn giúp cho cán bộ quản trị và cán bộ ở các phòng ban liên quan có thể tìm kiếm các thông tin trong dữ liệu log một cách nhanh chóng và tiện lợi nhất vì khi đó toàn bộ dữ liệu log được lưu trữ tập trung và đánh chỉ mục, khác với việc lưu trữ phân tán trước đây. Ngoài ra, với thông tin được lưu trữ tập trung trong Elasticsearch nên cán bộ quản trị có thể sử dụng để tổng hợp, thống kê tình hình hoạt động cũng như các vấn đề xảy ra trong tuần, tháng, quý,... việc này sẽ không thể thực hiện được với cách thức quản lý dữ liệu log thủ công truyền thống tại Bảo Việt. Ví dụ sau, từ dữ liệu log được đánh chỉ mục trong ElasticSeach chúng tôi có thể dễ dàng xây dựng biểu đồ thống kê tình trạng lỗi gặp phải của cơ sở dữ liệu theo thời gian thực:



Hình 3.6 : Biểu đồ thống kê mã lỗi gặp phải với hệ thống BVCare

Trường hợp thử nghiệm thứ 2, chúng tôi thực hiện giám sát và phân tích dữ liệu log của các kết nối vào cơ sở dữ liệu để kịp thời phát hiện ra các bất thường xảy ra đối với cơ sở dữ liệu của chúng tôi. Chúng tôi giả định trường hợp cơ sở dữ liệu có một kết nối bất thường từ một địa chỉ lạ, điều này có thể dẫn đến việc mất mát dữ liệu mà chúng tôi sẽ không thể kiểm soát được nếu không có hệ thống quản lý dữ liệu log. Kết quả khi có thông tin log ghi nhận có địa chỉ lạ kết nối tới cơ sở dữ liệu, lập tức hệ thống quản lý log sẽ thực hiện gửi email cảnh báo tới cán bộ quản trị:



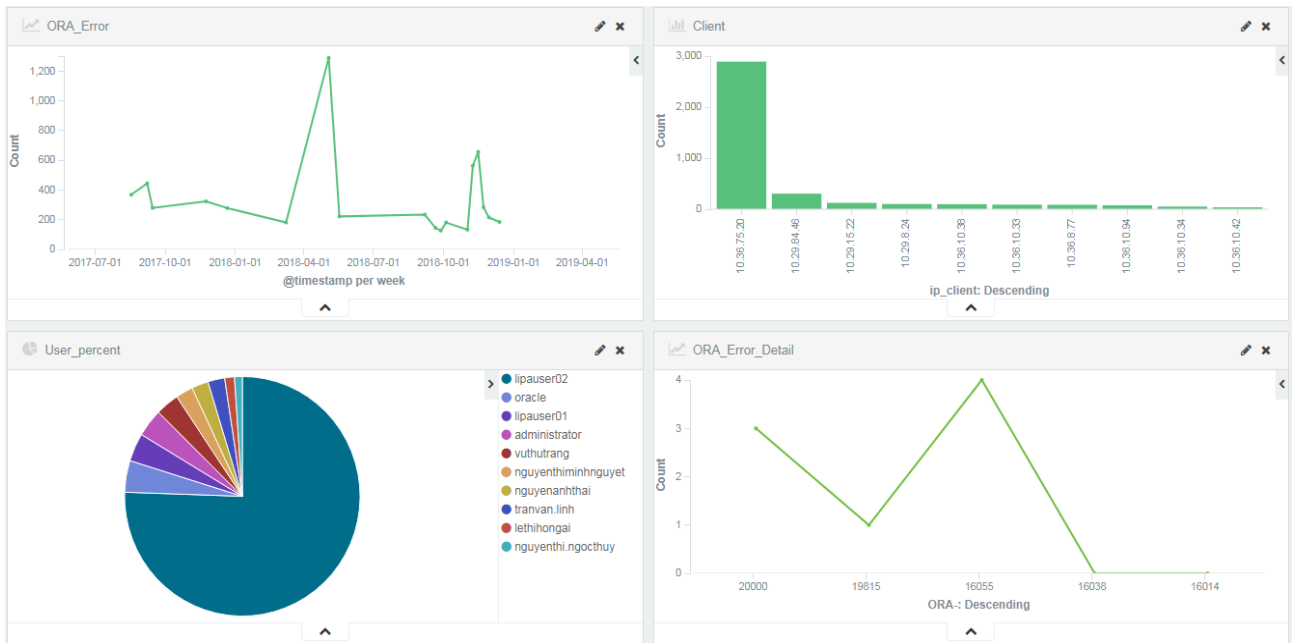
Cảnh Báo: Có địa chỉ lạ kết nối tới cơ sở dữ liệu BVCare  
Linh Tran đến: tranvan.linh

[Hiện chi tiết](#)

```
26-FEB-2019 09:02:29 * (CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=PDBTAL4DM)(CID=(PROGRAM=C:\Program?Files??x86?PLSQL?
Developer\plsqldev.exe)(HOST=ITCL2-THAINA)(USER=nguyenanhtai))) * (ADDRESS=(PROTOCOL=tcp)(HOST=10.36.10.94)(PORT=57561)) * establish * PDBTAL4DM
* 0
```

Hình 3.7 : Email cảnh báo địa chỉ lạ kết nối tới cơ sở dữ liệu hệ thống BVCare

Đồng thời, dữ liệu log được đánh chỉ mục trong Elasticsearch một cách thời gian thực (real-time) giúp chúng tôi có thể dễ dàng xây dựng các màn hình thống kê, biểu đồ để phân tích số liệu trên Kibana, tìm ra chính xác vấn đề ngay tại thời điểm hệ thống ghi nhận thông tin bất thường vào dữ liệu log, với sức mạnh của hệ truy hồi thông tin Elasticsearch kết hợp với phần mềm Kibana chúng tôi có thể thực hiện các công việc này một cách dễ dàng và nhanh chóng, điều mà sẽ không thể thực hiện được khi sử dụng phương pháp quản lý dữ liệu log thủ công truyền thống. Ví dụ dưới đây, chúng tôi thống kê danh sách các địa chỉ có kết nối tới cơ sở dữ liệu trong vòng 7 ngày trở lại, và các user được sử dụng để đăng nhập vào hệ thống:



Hình 3.8 : Biểu đồ thống kê địa chỉ và người dùng kết nối tới cơ sở dữ liệu BVCare

Từ biểu đồ trên chúng tôi có thể dễ dàng nhận ra xuất phát từ đâu có nhiều kết nối tới cơ sở dữ liệu nhất, và địa chỉ nào là bất thường, địa chỉ nào không thuộc diện được phép truy cập vào hệ thống nhưng bằng cách nào đó vẫn đang cố ý thực hiện kết nối vào hệ thống hay có thể biết được người dùng nào đang truy cập bất hợp pháp vào cơ sở dữ liệu của hệ thống,...

### 3.4. Đánh giá kết quả

Sau khi thực hiện cấu hình thử nghiệm dựa trên mô hình thử nghiệm được trình bày ở trên, chúng tôi đánh giá kết quả đạt được đã có thể giải quyết được 2 vấn đề trong bài toán quản lý dữ liệu log tại Bảo Việt như đã đề cập trong Chương 1 – mục 1.2: “Giới thiệu bài toán”, đó là:

- Quản lý được dữ liệu log tập trung trong Elasticsearch giúp dễ dàng tra cứu với hiệu năng cao, làm nguồn dữ liệu cho các bài toán phân tích, xây dựng các màn hình giám sát trên Kibana theo thời gian thực.
- Phát hiện và gửi thư điện tử cảnh báo lỗi, các bất thường xảy ra với hệ thống một cách tự động đến cán bộ quản trị mang tính chính xác cao và tức thời, giúp cán bộ kịp thời phát hiện và khắc phục lỗi trước khi xảy ra sự cố với hệ thống. So với công việc trước đây phải kiểm tra và tìm mã lỗi một cách thủ công thì kết quả này là một bước tiến dài phục vụ rất hữu ích cho doanh nghiệp, giải phóng được nguồn nhân lực thực hiện công việc thủ công.

Việc giải quyết được 2 vấn đề trên cũng đồng nghĩa với việc đã giải quyết được các hạn chế khi sử dụng cách thức quản lý và giám sát dữ liệu log truyền thống tại Bảo

Việt như đã được đề cập đến trong Chương 2 – mục 2.1.5: “Hiện trạng quản lý, giám sát hệ thống”.

Ngoài ra, với dữ liệu log được lưu trữ tập trung trong Elasticsearch và phần mềm Kibana mạnh mẽ, chúng tôi có thể xây dựng các biểu đồ, trừu tượng hóa dữ liệu theo nhiều chiều để phân tích và phát hiện các bất thường xảy ra với hệ thống trong thời gian thực, từ đó giúp nâng cao hiệu quả giám sát hệ thống và chất lượng dịch vụ mà chúng tôi cung cấp.

### **3.5. Các vấn đề còn tồn tại và hướng phát triển**

Qua quá trình thử nghiệm giải pháp trên, chúng tôi nhận thấy còn tồn tại một số vấn đề trong mô hình kiến trúc giải pháp đã được chúng tôi trình bày trong mục 2.3 của Chương II, đó là:

#### **3.5.1. Vấn đề sử dụng nhiều tài nguyên máy chủ**

Do Logstash được viết trên nền tảng ngôn ngữ Java, nên khi Logstash chạy nó sẽ cần cấp phát một máy chủ ảo Java (JVM), máy chủ ảo này sẽ chiếm một lượng bộ nhớ nhất định, khi có nhiều tệp dữ liệu log cần giám sát thì số lượng máy chủ JVM sẽ tăng theo và chiếm một phần lớn dung lượng bộ nhớ trong của máy chủ dịch vụ, điều này gây nên ảnh hưởng đến hiệu năng của dịch vụ.

#### **3.5.2. Vấn đề thất lạc dữ liệu log**

Với mô hình đang thử nghiệm, quá trình đẩy dữ liệu log từ Logstash đến Elasticsearch và tới mail server (máy chủ thư điện tử) là qua giao thức mạng, do đó, vì một lý do nào đó, tại thời điểm dữ liệu log được truyền đi mà đường mạng có vấn đề về kết nối thì phần dữ liệu log đó sẽ bị thất lạc và không được truyền đi. Đây là một rủi ro cần được kiểm soát để đảm bảo hệ thống giám sát và cảnh báo lỗi trong dữ liệu log hoạt động một cách đúng đắn và đầy đủ nhất, không bỏ sót bất kỳ sự kiện nào trong dữ liệu log.

#### **3.5.3. Hướng phát triển giải quyết vấn đề**

Qua quá trình nghiên cứu, chúng tôi nhận thấy 2 vấn đề gặp phải được trình bày trong mục 3.5.1 và 3.5.2 có thể được khắc phục, và đây cũng là hướng phát triển tiếp theo cho hệ thống quản lý dữ liệu log mà chúng tôi xây dựng.

➤ Với vấn đề Logstash sử dụng nhiều tài nguyên bộ nhớ của máy chủ chứa dữ liệu log được giải quyết theo phương án sau:

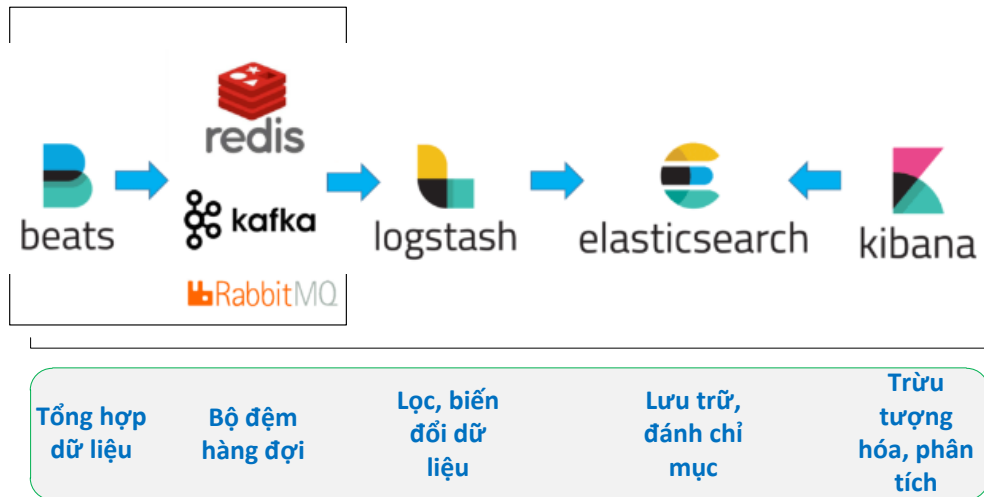
- Sử dụng Filebeat để tổng hợp dữ liệu log thay vì Logstash
- Logstash được sử dụng với vai trò lọc, chế biến và đẩy dữ liệu log nhận được từ Filebeat vào lưu trữ trong Elasticsearch. Filebeat được phát triển trên nền tảng Go API, nên Filebeat khá nhẹ, chạy tiết kiệm tài nguyên và hỗ trợ rất nhiều loại dữ liệu log và nền tảng khác nhau. Kết hợp sử dụng Filebeat và Logstash sẽ giải quyết tốt vấn đề sử



dụng nhiều tài nguyên gây ảnh hưởng đến hoạt động dịch vụ của Logstash.

➤ Với vấn đề thất lạc dữ liệu log có thể được giải quyết bằng cách sử dụng thêm thành phần bộ đệm hàng đợi (queue) để lưu giữ dữ liệu log đang chờ được xử lý. Có thể kể đến một số nền tảng bộ đệm hàng đợi nổi trội như Redis, Kafka hay RabbitMQ

Mô hình tổng thể hướng phát triển của giải pháp như sau:



Hình 3.9 : Mô hình hướng phát triển của giải pháp

### 3.6. Kết luận

Trong Chương III, luận văn đã trình bày chi tiết công việc triển khai thử nghiệm trên một hệ thống của Tập đoàn Bảo Việt, giải pháp quản lý log tập trung giúp người quản trị hệ thống có thể tiếp nhận được thông tin lỗi của hệ thống từ dữ liệu log một cách tức thời thông qua việc gửi thư điện tử cảnh báo tự động, từ đó người quản trị hệ thống có thể khắc phục lỗi kịp thời trước khi hệ thống gặp sự cố, gây gián đoạn dịch vụ. Từ nguồn dữ liệu log được tổng hợp tập trung vào hệ truy hỏi thông tin mạnh mẽ ElasticSearch còn là nguồn dữ liệu phục vụ để chúng tôi xây dựng các màn hình giám sát hệ thống, phân tích dữ liệu log để phân tích hành vi người dùng, các vấn đề liên quan đến an ninh bảo mật của hệ thống.

## KẾT LUẬN

Quản lý tốt dữ liệu log và có cơ chế cảnh báo tự động sẽ giúp cán bộ quản trị hệ thống sớm biết được các vấn đề xảy ra với hệ thống của mình theo thời gian thực để có thể thực hiện các biện pháp khắc phục kịp thời trước khi sự cố hệ thống thực sự xảy ra. Luận văn đã có một số đóng góp sau:

➤ **Thứ nhất:** đã phân tích được các điểm yếu tồn tại trong cách thức quản lý và giám sát dữ liệu log hiện tại của Tập đoàn Bảo Việt:

- Cán bộ phải truy xuất dữ liệu thủ công và phân tán trên các máy chủ để đọc và tìm mã lỗi.
- Việc tìm kiếm thủ công sẽ rất chậm và có thể gây ra nhầm lẫn hoặc thiếu sót trong quá trình kiểm tra dữ liệu log, gây rủi ro sẽ xảy ra sự cố cho hệ thống mà người quản trị không được biết kịp thời.
- Dữ liệu log không được tận dụng để xây dựng các báo cáo và phân tích để tìm ra những thông tin hữu ích.
- Mất nhiều nguồn lực con người để thực hiện các công việc đọc và tìm kiếm lỗi thủ công

➤ **Thứ hai:** xác định được 2 vấn đề chính cần giải quyết cho bài toán quản lý dữ liệu log tại Bảo Việt để có thể giải quyết được những điểm yếu, hạn chế trên, đó là:

- Quản lý được dữ liệu log tập trung giúp dễ dàng tra cứu với hiệu năng cao, làm nguồn dữ liệu cho các bài toán phân tích, xây dựng các màn hình thống kê, giám sát hệ thống theo thời gian thực.
- Phát hiện và gửi thư điện tử cảnh báo lỗi, các bất thường xảy ra với hệ thống một cách tự động đến cán bộ quản trị mang tính chính xác cao và tức thời, giúp cán bộ kịp thời phát hiện và khắc phục lỗi trước khi xảy ra sự cố với hệ thống.

➤ **Thứ ba:** đã phân tích, thiết kế và đưa vào thử nghiệm thành công giải pháp quản lý dữ liệu log sử dụng công nghệ ELK giúp giải quyết được 2 vấn đề đã phân tích ở trên, kết quả đạt được có thể hỗ trợ phát hiện sớm các vấn đề phát sinh bên trong hệ thống giúp cán bộ quản trị khắc phục kịp thời các lỗi có thể gây xảy ra sự cố cho hệ thống, giải pháp ELK giúp:

- Quản lý dữ liệu log tập trung trong hệ truy hồi thông tin Elasticsearch.
- Thu thập và phân tích dữ liệu log thời gian thực với Logstash.
- Gửi email cảnh báo thời gian thực ngay khi log ghi nhận vấn đề.
- Xây dựng các màn hình thống kê, giám sát thời gian thực với Kibana giúp cán bộ có thể theo dõi được hiện trạng hoạt động của hệ thống.
- Giải phóng nguồn nhân lực thu thập và tìm kiếm mã lỗi, kiểm tra hệ thống thủ công trước đây.

### **Vấn đề còn tồn tại**

Bên cạnh các kết quả đã đạt được, Luận văn còn một số hạn chế cần khắc phục trong thiết kế giải pháp như:

- Vấn đề Logstash sử dụng tốn dung lượng bộ nhớ máy chủ, ảnh hưởng đến hiệu năng của ứng dụng.
- Vấn đề rủi ro thất lạc dữ liệu log khi xảy ra lỗi đường truyền từ Logstash tới hệ truy hồi thông tin Elasticsearch.
- Vấn đề bảo mật với Kibana, hiện tại giao diện Kibana mã nguồn mở không hỗ trợ phân quyền người dùng đăng nhập hệ thống. Đây là vấn đề liên quan đến bảo mật dữ liệu log hệ thống cần được giải quyết.

### **Hướng phát triển kế tiếp**

Với những hạn chế đã phân tích ở trên, chúng tôi sẽ tiếp tục thực hiện nghiên cứu và hoàn thiện mô hình giải pháp, bước đầu chúng tôi đã tìm hiểu được các giải pháp có thể sử dụng để khắc phục các vấn đề còn tồn tại:

- Vấn đề Logstash sử dụng tốn dung lượng bộ nhớ máy chủ: Sử dụng Filebeat để thu thập dữ liệu log.
- Vấn đề rủi ro thất lạc dữ liệu log: Sử dụng bộ đệm hàng đợi Redis
- Vấn đề bảo mật với Kibana: Sử dụng Nginx làm lớp phân quyền người dùng đăng nhập vào hệ thống.

## TÀI LIỆU THAM KHẢO

### 1. Tài liệu tiếng việt

[1] PGS. TS. Nguyễn Trí Thành, *Bài giảng môn học các hệ truy hồi thông tin*, Đại học Công Nghệ.

### 2. Tài liệu tiếng anh

[2] Mounia Lalmas (2011), *Introduction to Information Retrieval*.

[3] Djoerd Hiemstra, *Information Retrieval Models*.

[4] Clinton Gormley and Zachary Tong (2015), *Elasticsearch: The Definitive Guide*.

[5] Patrick Ziegler and Klaus R. Dittrich (2007), “*Data Integration-Problems, Approaches, and Perspectives*”, Database Technology Research Group.

[6] Colin White, IBM BI Research(2006), *A roadmap to enterprise data integration*.

### 3. Các trang Web

[7] [https://en.wikipedia.org/wiki/Information\\_retrieval](https://en.wikipedia.org/wiki/Information_retrieval)

[8] [https://en.wikipedia.org/wiki/Vector\\_space\\_model](https://en.wikipedia.org/wiki/Vector_space_model)

[9] <https://www.elastic.co/guide/en/logstash/current/index.html>

[10] <http://www.oaktable.net/content/auditing-oracle-database-stopping-and-starting-using-elk-stack>

[11] <https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html>

[12] <https://www.elastic.co/guide/en/kibana/current/index.html>

[13] [https://en.wikipedia.org/wiki/Boolean\\_model\\_of\\_information\\_retrieval](https://en.wikipedia.org/wiki/Boolean_model_of_information_retrieval)