HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

ĐỀ TÀI:

TRIỂN KHAI XÂY DỰNG HỆ THỐNG GIÁM SÁT AN TOÀN MẠNG DỰA TRÊN CÁC PHẦN MỀM MÃ NGUỒN MỞ

Giảng viên hướng dẫn	**:**	**TS. Nguyễn Ngọc Điệp**
Sinh viên thực hiện	**:**	**Nguyễn Quang Hưng**
Lớp	**:**	**D13ATT2**
Khóa	**:**	**2013 - 2018**
Hệ	**:**	**Đại học chính quy**

HÀ NỘI – 2017

LỜI CẢM ƠN

Trước tiên, em xin chân thành cảm ơn các thầy, cô trong Khoa Công Nghệ Thông Tin và toàn thể các cán bộ của Học viện Công nghệ Bưu chính Viễn thông Hà Nội đã tận tình dạy dỗ, quan tâm, giúp đỡ, truyền đạt những kiến thức, kinh nghiệm cho em trong hơn 4 năm học tập tại đây.

Em xin tỏ lòng biết ơn sâu sắc đến thầy giáo TS. Nguyễn Ngọc Điệp, người đã tận tâm chỉ dẫn, định hướng cho em trong suốt quá trình học tập và làm đồ án tốt nghiệp này. Thầy luôn cho em những lời khuyên quý báu, giúp em giải quyết những vấn đề khó khăn trong quá trình nghiên cứu với tất cả lòng nhiệt huyết và tận tình để em có thể hoàn thành đồ án tốt nghiệp một cách tốt nhất. Xin gửi lời cảm ơn đến tập thể lớp D13ATTT, những người bạn đã luôn bên cạnh tôi những lúc khó khăn, động viên, giúp đỡ tôi trong quá trình học tập và thực hiện đồ án.

Cuối cùng xin gửi lời cảm ơn cha mẹ, gia đình và bạn bè. Những người đã luôn quan tâm, chăm sóc, ủng hộ và là nguồn động viên, là chỗ dựa vững chắc cho bản thân em trong suốt thời gian học tập để được như ngày hôm nay.

Xin chân thành cảm ơn!

Hà Nội, tháng 12 năm 2017

Sinh viên thực hiện

Nguyễn Quang Hưng

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT

Chữ viết tắt	Nghĩa tiếng anh	Nghĩa tiếng việt
ELK	Elasticsearch – Logstash - Kibana	Bộ công cụ quản lý và phân tích log
IDS	Intrusion detection system	Hệ thống phát hiện xâm nhập
IPS	Intrusion Prevention System	Hệ thống ngăn ngừa xâm nhập
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
JDBC	Java Database Connectivity	API tương tác với các loại cơ sở dữ liệu
SSL	Secure Sockets Layer	Giao thức bảo mật tầng mạng
TLS	Transport Layer Security	Giao thức bảo mật tầng giao vận
TCP	Transmission Control Protocol	Giao thức điều khiển truyền vận
JDK	Java Development Kit	Bộ công cụ phát triển Java
JSON	JavaScript Object Notation	Một kiểu dữ liệu mở trong JavaScript
UDP	User Datagram Protocol	Giao thức điều khiển truyền vận
API	Application Programming Interface	Phương thức kết nối với các thư viện và ứng dụng khác

DANH MỤC CÁC HÌNH ẢNH

MỞ ĐẦU

Ngày nay, với tốc độ phát triển chóng mặt của ngành công nghệ thông tin với nhu cầu quá lớn từ nhiều người sử dụng, dẫn đến các hệ thống máy tính trở nên phức tạp, khó quản lý. Nhiều hệ thống không phải ở cùng một nơi, nằm phân tán, các hệ điều hành, ứng dụng, dịch vụ được tạo ra bởi rất nhiều nguồn khác nhau. Lượng dữ liệu khổng lồ không ngừng gia tăng nhưng lại không tập trung. Vì vậy, chúng ta cần phải ghi lại hoạt động của hệ thống mà ở đây là Log. Thông thường, các bản ghi được lưu trữ phân tán trên các thiết bị khác nhau. Kiểm tra log theo phương pháp truyền thống là đăng nhập vào từng hệ thống để tìm kiểm tra, tìm kiếm lỗi gây mất thời gian, kém hiệu quả. Trong vai trò là người quản trị hệ thống hay là một chuyên gia bảo mật thông tin thì công tác giám sát luôn là việc cần thiết. Chúng ta cần phải biết những gì đang xảy ra trên hệ thống của mình vào mọi lúc mọi nơi. Nắm bắt mọi thông tin lịch sử về sử dụng, hiệu suất và tình trạng của tất cả các ứng dụng, thiết bị và tất cả dữ liệu trên hệ thống mạng. Chính vì vậy việc giám sát hệ thống là một công việc vô cùng quan trọng và cấp thiết đối với mọi tổ chức, doanh nghiệp và cơ quan.

ĐỒ ÁN SẼ GỒM 3 CHƯƠNG CHÍNH SAU:

Chương 1 - Tổng quan về giám sát an toàn mạng: Nội dung của chương này sẽ giới thiệu tổng quan về giám sát an toàn mạng, đưa ra mô hình và cách thức hoạt động chung của một hệ thống giám sát an toàn mạng. Tìm hiểu một số các công cụ hỗ trợ hệ thống giám sát an toàn mạng

Chương 2 – Đề xuất giải pháp: Nội dung của chương này sẽ đưa ra các giải pháp tương ứng cho từng thành phần của một hệ thống giám sát an toàn mạng. Đưa ra được mô hình tổng quan và cách thức hoạt động của các thành phần được lựa chọn.

Chương 3 - Triển khai hệ thống phần mềm: Nội dung của chương này tập trung vào việc áp dụng giải pháp đã được đề xuất ở chương II để xây dựng một hệ thống giám sát an toàn mạng nhằm theo dõi, giám sát và đưa ra được thông số cụ thể để có thể theo dõi được tình trạng của các máy tính ở trong hệ thống.

Kết luận: Tổng kết lại toàn bộ những công việc đã thực hiện trong đồ án này. Dựa trên những kết quả đạt được đề ra hướng nghiên cứu và phát triển tiếp theo trong tương lai.

CHƯƠNG 1. TỔNG QUAN VỀ GIÁM SÁT AN TOÀN MẠNG

Chương này trình bày về các vấn đề sau:

- Giới thiệu tổng quan về giám sát an toàn mạng
- Mô hình và cách thức hoạt động của hệ thống
- Tìm hiểu về các công cụ hỗ trợ hệ thống giám sát an toàn mạng

1.1. Giới thiệu chung

Internet phát triển, sự kết nối trên toàn thế giới đang mang lại thuận tiện cho tất cả mọi người. nhưng bên cạnh đó nó cũng tiềm ẩn những nguy cơ đe dọa đến mọi mặt của đời sống xã hội. việc đánh cắp thông tin, truy cập hệ thống trái phép, tấn công từ chối dịch vụ... là nguy cơ mà người dùng Internet phải đương đầu.

Rất nhiều các giải pháp an ninh mạng đã được đưa ra và cũng đã có những đóng góp to lớn trong việc đảm bảo an toàn thông tin, ví dụ như: Firewall ngăn chặn những kết nối không đáng tin cậy, mã hóa làm tăng độ an toàn cho việc truyền dữ liệu, các chương trình diệt virus với cơ sở dữ liệu được cập nhật thường xuyên…

Tuy nhiên thực tế cho thấy chúng ta vẫn luôn thụ động trước các cuộc tấn công đặc biệt là các tấn công kiểu mới vì vậy yêu cầu đặt ra là cần có một hệ thống phát hiện và cảnh báo sớm trước các cuộc tấn công. Hệ thống phát hiện xâm nhập được xem như là một lựa chọn tối ưu.

1.2. Giám sát mạng

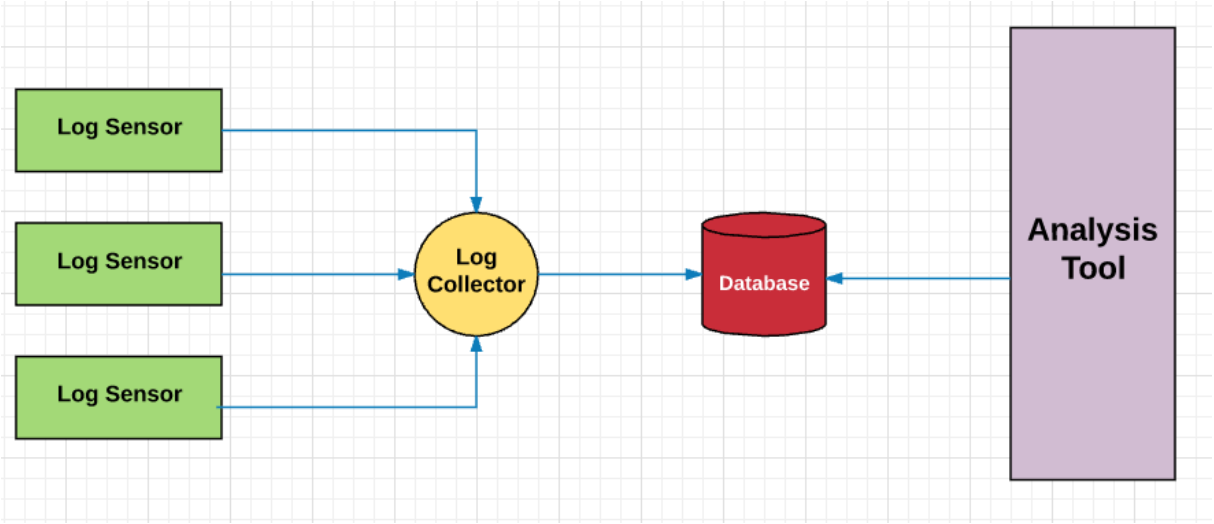
1.2.1. Khái niệm

Giám sát mạng là việc giám sát, theo dõi và ghi nhận những luồng dữ liệu mạng, từ đó sử dụng làm tư liệu để phân tích mỗi khi có sự cố xảy ra. Trong các hệ thống thông tin, việc khắc phục các sự cố thường tốn một chi phí rất lớn. Vì vậy, giải pháp giám sát mạng để phát hiện sớm các sự cố là một sự lựa chọn được nhiều người ưa thích nhằm mang lại hiệu quả cao với chi phí vừa phải.

1.2.2. Mô hình chung

Một nguyên tắc của thu thập dữ liệu là càng lấy được nhiều dữ liệu càng tốt, do các kiểu tấn công là đa dạng nên ta không thể biết được dữ liệu nào cần thiết, dữ liệu nào là không. Trên thực tế, một hệ thống mạng của doanh nghiệp/tổ chức bao gồm rất nhiều thành phần: web server, mail server, DNS server, file sharing system,…mỗi thành phần đều có hệ thống log của các ứng dụng chạy trên nó. Tất cả các dữ liệu này, nếu ta tiến hành phân tích dữ liệu tại chỗ đôi khi sẽ không đầy đủ thông tin để phát hiện tấn công và rất khó quản lý. Mặt khác, khi log file lớn, việc phân tích sẽ tiêu tốn

tài nguyên và ảnh hưởng tới hiệu năng của chính máy chủ hiện tại. Vì vậy, phân tích dữ liệu tập trung và phân tách quá trình phân tích ra khỏi hoạt động của hệ thống sẽ là một hướng đi đúng đắn hơn. Giải pháp đề xuất ở đây bao gồm các thành phần chính sau: Log Sensor, Log Collector, Database và Analysis tool.



Hình 1. 1: Sơ đồ tổng quan về mô hình giám sát mạng

Một hệ thống giám sát mạng thường có các thành phần chính sau:

- Máy trinh sát (Log Sensor): là những máy trạm làm nhiệm vụ trinh sát. Thành phần này sẽ tiếp cận, tương tác với các hệ thống và dịch vụ cần giám sát để nhận biết trạng thái của những dịch vụ đó. Trong quá trình triển khai hệ thống, thành phần này sẽ được phân tán nằm rải rác nhiều nơi trên mạng để thu thập thông tin từ những nguồn khác nhau như tường lửa, bộ định tuyến, tập tin nhật ký…
- Máy thu thập (Log Collector): Một điều đáng chú ý trong hệ thống giám sát mạng là các hệ thống, các dịch vụ cần giám sát có thể khác nhau. Điều này đồng nghĩa với việc thông tin thu được cũng có nhiều định dạng khác nhau. Để có được thông tin một cách đồng nhất nhằm mục đích xử lý và thống kê, cần có một thành phần làm nhiệm vụ chuẩn hóa thông tin. Máy thu thập sẽ đọc những thông tin thu được từ các máy trinh sát và chuẩn hóa thông tin dựa trên những quy tắc chuẩn hóa biết trước. Thông tin đầu ra sẽ có định dạng giống nhau và được lưu vào cơ sở dữ liệu trung tâm.
- Cơ sở dữ liệu trung tâm: là nơi lưu trữ dữ liệu của toàn bộ hệ thống giám sát. Các dữ liệu ở đây đã được chuẩn hóa nên có thể sử dụng để tính toán các số liệu thống kê trên toàn hệ thống

- Công cụ phân tích (Analysis tool): Thành phần này sẽ đọc các dữ liệu từ cơ sở dữ liệu trung tâm và tính toán để tạo ra bản báo cáo số liệu thống kê trên toàn hệ thống.

1.3. Cách thức hoạt động và mục đích của từng thành phần

1.3.1. Thu thập dữ liệu

Việc thu thập dữ liệu ở đây chính là việc lấy các thông tin liên quan đến tình trạng hoạt động của các thiết bị trong hệ thống mạng. Tuy nhiên, trong những hệ thống mạng lớn thì các dịch vụ hay các thiết bị không đặt trên máy, một địa điểm mà nằm trên các máy chủ, các hệ thống con riêng biệt nhau. Các thành phần hệ thống cũng hoạt động trên những nền tảng hoàn toàn khác nhau. Mô hình log tập trung được đưa ra để giải quyết vấn đề này. Cụ thể, là tất cả log sẽ được chuyển về một trung tâm ở đây là Log Collector để phân tích và xử lý.

Với mỗi thiết bị có những đặc điểm riêng và các loại log cũng khác nhau. Như log của các thiết bị mạng như: Router, Swich. Log của các thiết bị phát hiện xâm nhập: IDS, IPS, Snort … Log của các Web Server, Application Server, Log Event, Log Registry của các Server Windows, Unix/Linux.

1.3.2. Phân tích dữ liệu

Khi đã thu thập được những thông tin về hệ thống thì công việc tiếp theo là phân tích thông tin, cụ thể là việc thực hiện chỉ mục hóa dữ liệu, phát hiện những điều bất thường, những mối đe dọa của hệ thống. Dựa trên những thông tin về lưu lượng truy cập, trạng thái truy cập, định dạng request… Ví dụ như lưu lượng truy cập bỗng dưng tăng vọt tại một thời điểm.

Phương pháp đẩy: Các sự kiện từ các thiết bị, các máy trạm, máy chủ sẽ được tự động chuyển về các Log Collector theo thời gian thực hoặc sau mỗi khoảng thời gian phụ thuộc vào việc cấu hình trên các thiết bị tương ứng. Các Log Collector sẽ thực hiện việc nghe và nhận các sự kiện khi chúng xảy ra.

1.3.3. Phát hiện và phản ứng

Phát hiện và phản ứng là hai thành phần quan trọng trong các yếu tố của tiến trình. Sau khi bức tường phòng ngự cuối cùng bị phá vỡ, các tổ chức cần nhanh trong phát hiện ra cách thức xâm nhập của kẻ tấn công và chúng sẽ làm gì tiếp theo. Quá trình này được gọi là phạm vi ứng phó sự cố. Bởi xâm nhập không có nghĩa là có quyền root. Một kẻ xâm nhập có thể leo thang đặc quyền của mình để thực hiện những âm mưu sau đó.

Bất kỳ ai khi thực hiện công việc ứng phó sự cố thường xuyên sẽ hiểu được công việc nào nên làm trước vì giám đốc, CEO, hay những nhân viên cấp cao không quan tâm đến việc kẻ xâm nhập làm thế nào mà chỉ quan tâm đến những vấn đề sau:

- Những kẻ tấn công đã làm gì
- Khi nào
- Chúng ta ngăn chặn được chưa
- Đã có thiệt hại như thế nào

Mặc dù các nhà lãnh đạo không quan tâm đến cách thức xâm nhập của kẻ tấn công, nhưng đó luôn là công việc hàng đầu để có thể phản ứng hiệu quả với các cuộc xâm nhập. Chỉ có cách xác định phương thức xâm nhập của kẻ tấn công và ngăn chặn chúng thì việc phục hồi mới có thể diễn ra trọn vẹn được.

1.3.4. Cảnh báo

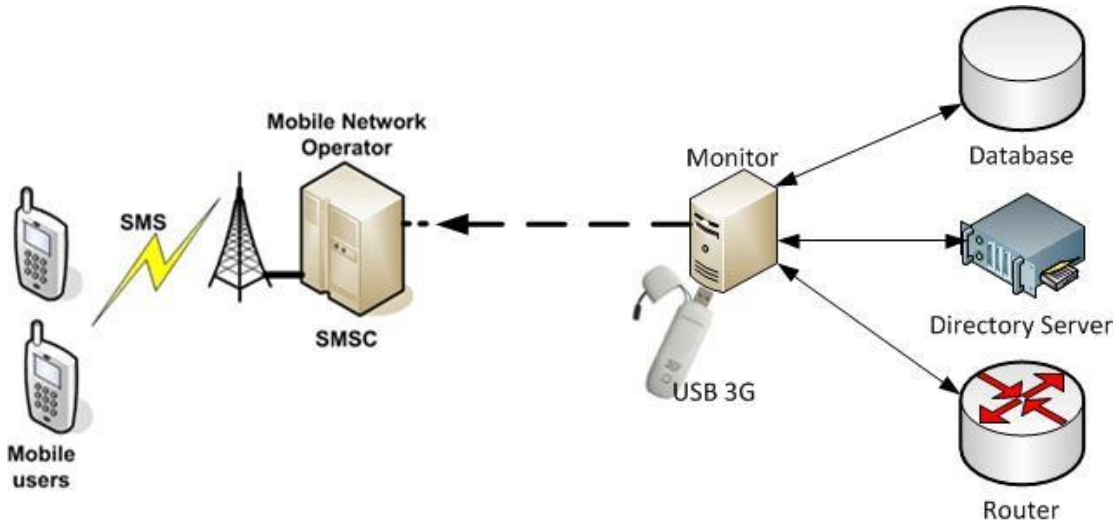
Sau khi đã thực hiện việc phân tích dữ liệu từ các thông tin thu thập bước tiếp theo là thực hiện việc đánh giá, đưa thông tin cảnh báo tới người quản trị và thực hiện những công tác nhằm chống lại những mỗi đe dọa, khắc phục các sự cố có thể sảy ra.

Cảnh báo có thể thông qua email, tin nhắn, hoặc thực thi các mã script nhằm hạn chế hậu quả của sự cố. Khi xảy ra sự cố, hệ thống sẽ tự động gửi email, tin nhắn cho người quản trị và cũng có thể chạy script để thêm một địa chỉ IP có biểu hiện tấn công vào trong danh sách đen của Firewall. Việc này đòi hỏi người lập trình phải có hiểu biết sâu và kinh nghiệm về hệ thống.

1.4. Tìm hiểu về một số công cụ hỗ trợ giám sát an ninh mạng

1.4.1. Nagios

Nagios là một hệ thống dùng để giám sát một hệ thống mạng. Nagios thực hiện việc theo dõi và đưa ra các cảnh báo về trạng thái các host và các dịch vụ. Nó được xây dựng trên nền Linux và đã hỗ trợ hầu hết các hệ điều hành tương tự Linux. Một điểm khác so với các công cụ khác là Nagios giám sát dựa tình trạng hoạt động của các máy trạm và dịch vụ. Nó sử dụng các Plug-in được cài đặt trên các máy trạm, thực hiện việc kiểm tra các máy trạm và dịch vụ theo định kỳ và gửi thông tin trạng thái về Nagios Server sau đó thông tin sẽ được đưa lên với một giao diện Web (Sử dụng Nagvis) và có thể gửi thông tin về trạng thái tới nhà quản trị qua email, tin nhắn… khi có sự cố xảy ra. Việc theo dõi có thể được cấu hình một cách chủ động hoặc bị động dựa trên mục đích sử dụng của người quản trị.



Hình 1. 2: Mô hình Nagios

- Ưu điểm:
  - Dễ dàng phát triển các plug-in riêng. Cho phép người sử dụng dễ dàng phát triển các dịch vụ giám sát nhu cầu sử dụng bằng việc sử dụng các ngôn ngữ shell script, C ++, Perl, Ruby, Python, PHP, C# ….).
  - Việc giám sát các dịch vụ là song song.
  - Thông tin cảnh báo (khi host và các dịch vụ xảy ra xự cố) bằng email, tin nhắn….
  - Sử dụng giao diện trên nền web để theo dõi trạng thái của mạng, xem lịch sử các cảnh báo và các sự cố xảy ra.
- Nhược điểm:
  - Nagios chỉ hoạt động trên các máy chủ chạy hệ điều hành họ Unix/Linux.
  - Việc truyền tải một lượng dữ liệu lớn cũng làm giảm hiệu suất trong việc phân tích và cảnh báo. Đặc biệt là với những mạng có tốc độ truy cập thấp.
  - Nagios không hỗ trợ các tính năng tự động khắc phục lỗi.

1.4.2. Syslog-Ng

Syslog-ng là một công cụ thu thập log rất hiệu quả và linh hoạt là sự lựa chọn của rất nhiều nhà quản trị mạng trong việc xây dựng một hệ thống log tập trung. Syslog-ng được xây dựng dựa trên chuẩn syslog trên nền tảng Unix và các hệ điều hành tương tự. Gồm xây dựng với 2 thành phần Syslog-ng client và Syslog-ng Server. Các Client thực hiện việc thu thập log quan trọng gửi tới máy chủ tập trung và lưu trữ.

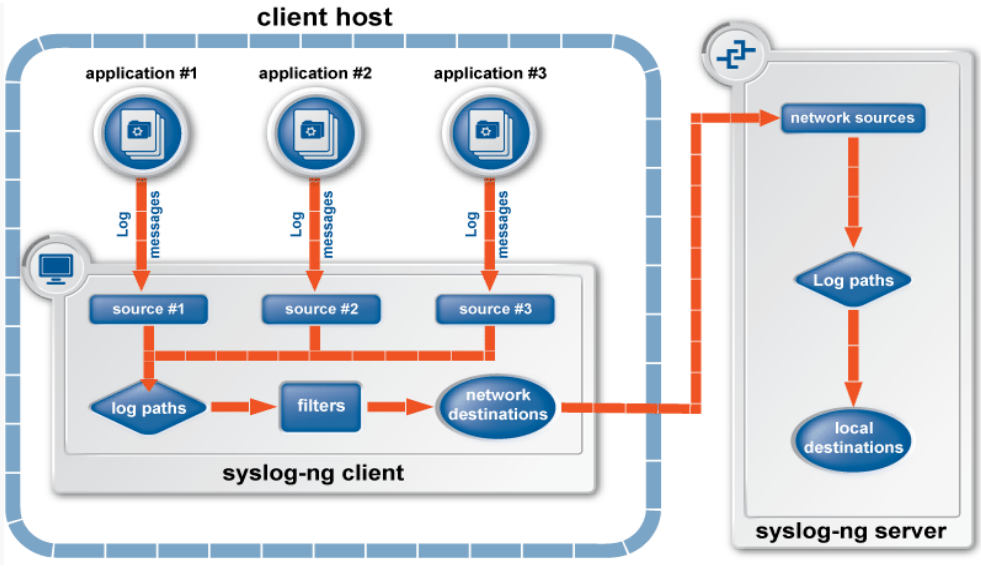
- Ưu điểm:

- Lưu trữ: Với Syslog-ng, ta có thể lưu trữ dữ liệu vào cơ sở dữ liệu cho phép tìm kiếm và truy vấn dễ dàng. Syslog-ng hỗ trợ các hệ cơ sở dữ liệu: MSSQL, MYSQL, Oracle và PostgreSQL.
- Lọc và phân loại: Syslog-ng cung cấp cơ chế lọc nhằm phân loại các Log message và cũng hạn chế lượng dữ liệu đổ về server log từ các client. Cơ chế lọc của Syslog-ng dựa trên các thông số khác nhau như source host, ứng dụng, sự ưu tiên trong log.
- Thu thập dữ liệu: Syslog-client thực hiện việc tập trung log từ các máy chủ và gửi về Syslog server. Syslog-ng thực hiện việc thu thập log từ các máy chủ khác nhau dựa trên giao thức TCP, đảm bảo không bị mất mát thông tin trên đường truyền Synlog-ng cung cấp một số cơ chế truy xuất log an toàn dựa trên SSL/TLS

- Nhược điểm:
  - Syslog-ng không phải là 1 phần mềm phân tích cho nên syslog-ng chỉ có thể lọc những log message phù hợp với 1 số tiêu chí định trước. Syslog-ng không thể làm tốt nhiệm vụ phân tích và cảnh báo các nguy cơ đến người quản trị.

1.4.3. Logzilla (Php Syslog-Ng)

Là phần mềm mã nguồn mở hỗ trợ việc quản lý log tập trung được phát triển dự trên PHP-Syslog-ng. Logzilla có thể quản lý với hàng triệu thông điệp log, hàng ngàn thiết bị cùng lúc. Được xây dựng trên nền web với một giao diện quản lý trực quan và thuận tiện cho người dùng. Là sự lựa chọn của nhiều nhà quản lý và giám sát anh ninh mạng.

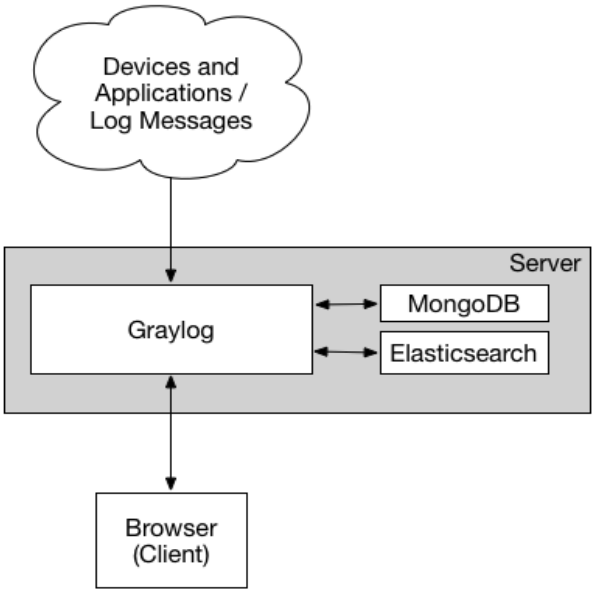


Hình 1. 3: Mô hình Logzilla

- Ưu điểm:
  - Tìm kiếm thông tin: Logzilla cung cấp một giao diện tìm kiêm theo từ khóa và theo một số thuộc tính khá trực quan và thông minh.
  - Cảnh báo và giám sát mạng: Logzilla hỗ trợ việc phát hiện các sự kiện một cách nhanh chóng trong thời gian thực. Có thể nhanh chóng phát hiện các điểm suy thoái của các thiết bị và máy chủ. Logzilla cũng hỗ trợ việc cảnh báo qua Email.
  - Trích xuất thông tin: Logzilla cũng hỗ trợ việc tạo kết xuất ra các báo cáo theo các định dạng: Excel và CSV. Logzilla còn hỗ trợ việc hiển thị dưới một số dạng biểu đồ giúp người quản trị dễ dàng hình dung hệ thống một cách trực quan.
- Nhược điểm:
  - Để triển khai Logzilla ta cần kết hợp với một hệ thống thu thập Log khác thực hiện công việc thu thập thông tin từ các máy chủ và thiết bị khác trên mạng.
  - Logzilla mặc định không hỗ trợ việc thu thập log từ các thiết bị hay các máy chủ khác nó tập trung vào việc thực hiện trên log đã có dựa trên việc thu thập log của Syslog-ng.

1.4.4. Graylog

Là một hệ thống quản lý Log mã nguồn mở, nó được xây dựng bằng Rubu trên Rails và MongoDB. Dựa trên một định dạng Log riêng dựa trên JSON-based được gọi là GELF (Graylog Extended Log Format).



Hình 1. 4: Mô hình Graylog

- Ưu điểm:
  - Có cơ chế nhận log rất linh hoạt: Có thể nhận log từ rất nhiều nguồn khác nhau.
  - Graylog có sử dụng Elasticsearch để lưu trữ và đánh dấu thông tin. Nên các việc tìm kiếm thông tin trong cơ sở dữ liệu sẽ rất chính xác, nhanh chóng và linh hoạt.
  - Có thể phân tích dữ liệu thành dạng số liệu thống kê, biểu đồ. Sau đó sẽ được hiển thị lên trình duyệt web để giúp các người quản trị dễ dàng theo dõi
  - Có các cơ chế cảnh báo linh hoạt. Ví dụ như cảnh báo qua Email, Slack…..
- Nhược điểm:
  - Nó chưa có cơ chế giúp tự động phát hiện ra các tấn công hay các vấn đề từ bên ngoài. Các điều này đều phải phụ thuộc vào kinh nghiệm của người quản trị hệ thống
  - Để phát huy hết được sức mạnh của Splunk cần có thời gian tìm hiểu và sử dụng. Nó chưa có cơ chế giúp tự động phát hiện ra các tấn công hay các vấn đề từ bên ngoài. Nhưng điều này phụ thuộc vào kinh nghiệm sử dụng và vốn hiểu biết của người quản trị.
  - Đề triển khai được một hệ thống sử dụng Splunk hiệu quả chúng ta cũng cần có một hệ thống riêng, đây cũng là một trở ngại không nhỏ với các hệ thống có quy mô trung bình và nhỏ.

- Do tất cả các module đều được tích hợp hết trong Graylog nên khi muốn mở rộng hệ thống sẽ gặp nhiều khó khăn.

1.4.5. ELK Stack

ELK stack được tạo lên từ 03 thành phần mã nguồn mở Elasticsearch, Logstash, Kibana có chức năng thu thập, phân tích, lưu trữ, tìm kiếm, hiển thị dữ liệu.

- Logstash: Đây là một công cụ sử dụng để thu thập, xử lý log được viết bằng java. Nhiệm vụ chính của logstash là thu thập log sau đó chuyển vào Elastichsearch. Mỗi dòng log của logstash được lưu trữ đưới dạng json.
- Elasticsearch: sử dụng cơ sở dữ liệu NoSQL dựa trên nền tảng của Apache Lucene engine. Dùng để lưu trữ dữ liệu và cung cấp interface cho phép truy vấn đến cơ sở dữ liệu.
- Kibana: Đây là giao diện sử dụng dành cho người dùng trên môi trường web. Kibana sẽ sử dụng Elashtichsearch để tìm kiếm các dữ liệu phù hợp với yêu cầu của người dùng.

- Ưu điểm:

  - Tìm kiếm thông tin: Kibana cung cấp một giao diện tìm kiêm theo từ khóa và theo một số thuộc tính khá trực quan và thông minh.
  - Trích xuất thông tin: Kibana cũng hỗ trợ việc tạo báo cáo ra các loại file khác khau. Kibana còn hỗ trợ việc hiển thị dưới một số dạng biểu đồ giúp người quản trị dễ dàng hình dung hệ thống một cách trực quan.
  - Đánh chỉ mục dữ liệu: Elasticsearch có thể đánh chỉ mục dữ liệu với một khối lượng dữ liệu lớn trong một khoảng thời gian ngắn. Giúp việc tìm kiếm diễn ra nhanh chóng và thuận tiện.
  - Tìm kiếm thông tin: Elasticsearch được xây dựng trên engine là Lucene, một thư viện full text search nên nó cung cấp cơ chế tìm kiếm cực kỳ thông minh bao gồm các từ khóa, các hàm và cấu trúc tìm kiếm giúp người sử dụng có thể truy xuất các thông tin.
  - Các hình thức thu thập dữ liệu: Logstash có thể thực hiện việc thu thập log từ rất nhiều nguồn khác nhau. Từ các file qua các kết nối UDP, TCP từ các Logstash Server khác , từ các Event Logs, Registry của Windows …Logstash kết hợp tốt với các công cụ thu thập log, các IDS cảnh báo khác.

- Nhược điểm:

- Do Logstash không có buffer nên việc truyền tải một lượng dữ liệu lớn sẽ làm giảm hiệu suất trong việc phân tích và cảnh báo. Đặc biệt là với những mạng có tốc độ truy cập thấp.
- Nó chưa có cơ chế giúp tự động phát hiện ra các tấn công hay các vấn đề từ bên ngoài. Các điều này đều phải phụ thuộc vào kinh nghiệm của người quản trị hệ thống

1.5. Phân tích khó khăn của những công cụ đã có

1.5.1. Nhược điểm của những công cụ

Từ những công cụ đã giới thiệu ở phần trên. Công cụ nào cũng có những ưu, nhược điểm nhất định, trong đó có thể kể đến một vài các nhược điểm điển hình của các công cụ như không hỗ trợ việc thu thập log từ nhiều các nguồn đầu vào khác nhau, không hỗ trợ việc hoạt động trên nhiều nền tảng hệ điều hành khác nhau hay là không có những tính năng có thể tự động khắc phục lỗi. Những nhược điểm này đều là những nhược điểm điển hình của các công cụ phân tích log, tuy nhiên thì nó cũng không ảnh hưởng nhiều đến tính khả dụng và hiệu suất làm việc của một hệ thống phân tích log. Mà ở đây có một vấn đề đáng quan tầm và nó cũng là một nhược điểm mà hầu như tất cả các công cụ, hệ thống phân tích log đều gặp phải là không có các hệ thống bộ nhớ đệm kèm theo và việc này có thể ảnh hưởng trực điểm đến tính khả dụng cũng như hiệu suất làm việc của một hệ thống.

Ở đây em có thể lấy ví dụ ngay với nhược điểm không có bộ nhớ đệm đối với công cụ phân tích Logstash. Trong Logstash, sự kiện được truyền từ module này tới module khác thông qua các hàng đợi sự kiện. Logstash đặt kích thước của các hàng đợi này là 20, nghĩa là có tối đa 20 sự kiện được chờ để được xử lý. Khi các hàng đợi này đầy, mọi thao tác ghi lên chúng đều bị block, một khi input gặp vấn đề, do ổ đĩa của đích đầy, nghẽn mạng, không có đủ quyền…Input sẽ tạm dừng và đợi cho tới khi nó có thể tiếp tục gửi tin nhắn. Điều đó có nghĩa là nó sẽ ngừng đọc dữ liệu từ hàng đợi, làm cho hàng đợi bị đầy, dẫn tới các hàng đợi khác cũng đầy, logstash sẽ tạm dừng đọc dữ liệu từ nguồn. Điều này sẽ không đảm bảo được tính thời gian thực của một hệ thống và khi các dữ liệu liên tục được đẩy tới hệ thống thì logstash sẽ không phân tích dữ liệu một cách kịp thời và sẽ dẫn đến các kết quả không tốt như ảnh hưởng trực tiếp đến kết quả của hệ thống giám sát an toàn mạng hay ảnh hưởng đến việc đưa ra kết luận sai lầm của người giám sát hệ thống mạng.

1.5.2. Hướng giải quyết

Để giải quyết vấn đề ảnh hưởng đến tính khả dụng và hiệu năng của hệ thống như ở phía trên, em xin đưa ra một đề xuất giải pháp là thiết kế các hàng đợi không giới hạn kích thước cho một hệ thống giám sát**.** Nhiệm vụ chính của hàng đợi này là đóng vai trò như một người thu thập dữ liệu này từ các nguồn khác nhau, tạo thành nhưng stream dữ liệu sau đó cung cấp lại cho các hệ thống khác để xử lý và nó cũng xử lý luôn được vấn đề khi mà một hệ thống bị quá tải và không xử lý kịp các yêu cầu thì nó sẽ có nhiệm vụ lưu lại các yêu cầu và gửi lại cho hệ thống khi hệ thống có thể tiếp tục xử lý. Do đóng vai trò là một trạm trung chuyển giữa các hệ thống với nhau nên hàng đợi cần đáp ứng một vài tiêu chí sau:

- Tốc độ nhanh
- Có khả năng mở rộng
- Độ tin cậy cao
- Có khả năng xử lý trong thời gian thực

Ở đây có thể kể đến một vài các hệ thống truyền thông điệp như RabbitMQ, ActiveMQ, ZeroMQ hay Kafka

- RabbitMQ: là một message broker (message-oriented middleware) sử dụng giao thức AMQP - Advanced Message Queue Protocol (Đây là giao thức phổ biến, thực tế rabbitmq hỗ trợ nhiều giao thức). RabbitMQ được lập trình bằng ngôn ngữ Erlang. RabbitMQ cung cấp cho lập trình viên một phương tiện trung gian để giao tiếp giữa nhiều thành phần trong một hệ thống lớn. RabbitMQ rất dễ sử dụng và triển khai, tuy nhiên nó rất khó mở rộng thêm mô hình và hiệu suất hoạt động rất chậm.
- ActiveMQ: là một messaging open source nổi tiếng và mạnh mẽ, ActiveMQ có thể chạy độc lập hay bên trong các tiến trình khác, ứng dụng server, hay ứng dụng JEE. Hỗ trợ mọi thứ JMS yêu cầu, và có thể mở rộng. Ngoài Java thì ActiveMQ có thể ứng dụng với .NET, C/C++, Ruby, Delphy.
- ZeroMQ: là một thư viện message (messaging library) thực thi các mẫu message phổ biến trong các những ứng dụng phân tán, là một system nhắn tin rất nhẹ được thiết kế đặc biệt cho các hệ thống yêu cầu độ trễ cực nhỏ như trong các hệ thống về ngân hàng và chứng khoán. Tuy nhiên ZeroMQ chỉ lưu trữ các message trong bộ đệm in-memory hết sức giới hạn và không hỗ trợ việc xem lại.
- Kafka: Apache Kafka là hệ thống truyền thông điệp phân tán, độ tin cậy cao, dễ dàng mở rộng và có thông lượng cao. Kafka cung cấp cơ chế offset (có

thể hiểu như tương tự như chỉ số của một mảng) để lấy thông điệp một cách linh hoạt, cho phép các ứng dụng xử lý có thể xử lý lại dữ liệu nếu việc xử lý trước đó bị lỗi. Ngoài ra, cơ chế "đăng kí" theo dõi cho phép việc lấy thông điệp ra gần như tức thời ngay khi dữ liệu đi vào hàng đợi. Kafka được thiết kế hỗ trợ tốt cho việc thu thập dữ liệu thời gian thực. Apache Kafka là hệ thống lưu trữ thông điệp được phát triển tại LinkedIn.

Giải pháp lựa chọn: Apache Kafka là một message broker, đặc biệt được sử dụng cho các đầu vào dữ liệu lớn. Các thiết kế làm cho Kafka thực sự là sự lựa chọn thích hợp nhất cho các tiến trình logging, lượng tài nguyên mà Kafka chiếm dụng cũng ít hơn nhiều so với các hệ thống truyền thông điệp khác. Do các yếu tố trên mà Kafka thích hợp hơn cho các ứng dụng xử lý theo thời gian thực với lượng dữ liệu lớn. Kafka có một vài những đặc điểm sau đây có thể đáp ứng được những yêu cầu của một hệ thống giám sát an toàn mạng:

- Tốc độ nhanh: Với một máy đơn cài đặt Kafka có thể xử lý số lượng dữ liệu từ việc đọc và ghi lên tới hàng trăm megabyte trong một giây từ hàng ngàn máy khách.
- Khả năng mở rộng: Kafka được thiết kế cho phép dễ dàng được mở rộng và trong suốt với người dùng (nghĩa là không có thời gian chết – ngừng hoạt động trong khi thêm một nút máy chủ mới vào cụm). Khi Kafka chạy trên một cụm, luồng dữ liệu sẽ được phân chia và được vận chuyển tới các nút trong cụm, do đó cho phép trung chuyển các dữ liệu mà có khối lượng lớn hơn nhiều so với sức chứa của một máy đơn.
- Độ tin cậy: Dữ liệu vào hàng đợi sẽ được lưu trữ trên ổ đĩa và được sao chép tới các nút khác trong cụm để ngăn ngừa việc mất dữ liệu, như vậy Kafka đảm bảo tính chịu lỗi cao.

1.6. Kết luận

Hiện nay, có rất nhiều công cụ hỗ trợ việc giám sát an ninh mạng. Tuy nhiên, mỗi công cụ, sản phẩm có những điểm mạnh, điểm yếu riêng đòi hỏi người quản trị cần có kinh nghiệm trong việc sử dụng sản phẩm. Lựa chọn công cụ, sản phẩm dựa trên các yêu cầu về quy mô của hệ thống, mức độ an toàn và nhiệm vụ của hệ thống cũng như kinh phí trong việc phát triển hệ thống.

Dựa trên mục tiêu nghiên cứu của đề tài là xây dựng một hệ thống giám sát an toàn mạng cho các công ty có quy mô vừa và nhỏ. Căn cứ vào ưu điểm, nhược điểm của các bộ công cụ đã nêu ở phía trên thì em xin đưa ra đề xuất lựa chọn bộ công cụ

mã nguồn mở ELK để xây dựng hệ thống giám sát an toàn mạng và lựa chọn công cụ mã nguồn mở Kafka có chức năng chính là truyền thông điệp.

Hệ thống giám sát an toàn mạng sau khi xây dựng thành công cần đáp ứng được một số các yêu cầu sau:

- Hệ thống giám sát an toàn mạng cần phải hoạt động ổn định cho các doanh nghiệp vừa và nhỏ.
- Hệ thống cần hiển thị đầy đủ và chính xác tất cả những hoạt động, sự kiện của các máy chủ được giám sát theo thời gian thực.
- Hệ thống sẽ hiển thị được các bảng biểu, số liệu thống kê trực quan giúp người quản trị hệ thống dễ dàng theo dõi. Ví dụ như:
  - Số lượng các request vào trong hệ thống theo thời gian thực
  - Top 10 các địa chi IP có số lượng request cao nhất
  - Top 5 các mã response được trả về nhiều nhất
  - Các tài nguyên của hệ thống như CPU, DISK, RAM đang được sử dụng.

1.7. Kết chương

Trong chương 1 đã giới thiệu tổng quan được về giám sát an toàn mạng, đưa ra được mô hình và cách thức hoạt động chung của một hệ thống giám sát an toàn mạng. Tìm hiểu về ưu điểm và nhược điểm của một số các công cụ hỗ trợ hệ thống giám sát an toàn mạng để từ đó đưa ra được các đề xuất lựa chọn các công cụ phù hợp cho việc xây dựng mô hình.

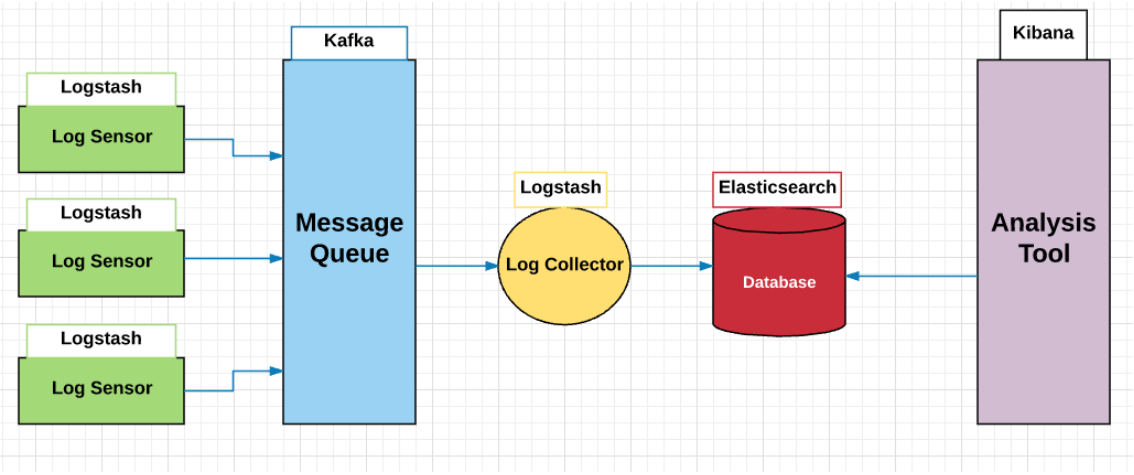
CHƯƠNG 2. ĐỀ XUẤT GIẢI PHÁP

Chương này trình bày về các vấn đề sau:

- Giới thiệu mô hình tổng quan
- Hoạt động của từng thành phần
- Giới thiệu về các thành phần phi chức năng

2.1. Mô hình tổng quan

Từ những thành phần cốt lõi đã được lựa chọn cho việc xây dựng mô hình giám sát an toàn mạng. Các thành phần đã được lựa chọn là bộ công cụ mã nguồn mở ELK phục vụ cho việc phân tích và giám sát hệ thống, bộ công cụ mã nguồn mở Kafka phục vụ cho việc truyền thông điệp trong hệ thống. Thì từ những thành phần đã được lựa chọn thì em xin đưa ra đề xuất mô hình tổng quan cho hệ thống giám sát an toàn mạng. Mô hình sẽ có các thành phần như ở hình 2.1:



Hình 2. 1: Mô hình tổng quan

2.2. Hoạt động của từng thành phần

2.2.1. Log Sensor

- Nhiệm vụ: thu thập tập tin lưu trữ log từ mọi nguồn trong hệ thống.
- Yêu cầu:
  - Đa nền tảng: do các tập tin log có ở khắp nơi, trên nhiều các nền tảng khác nhau, theo nhiều các tiêu chuẩn log khác nhau nên Log Sensor cần phải đáp ứng được việc hỗ trợ các nền tảng hệ điều hành khác nhau và tiêu chuẩn các tập tin log hiện nay.

- Thời gian thực: Log Sensor cần phải theo dõi được sự thay đổi của các tập tin log ngay tức thì để chuyển các tập tin log sang message queue, nhằm giảm tối đa độ trễ trong việc phân tích và đưa ra kết quả.
- Dữ liệu được lưu trên các tập tin log thường ở dạng thô nên để thuận tiện cho quá trình lưu trữ và phân tích, Log Sensor phải đáp ứng được việc chuyển đổi các tập tin log này sang các kiểu dữ liệu có cấu trúc.

- Giải pháp lựa chọn: Logstash. Đây là phần mềm mã nguồn mở chuyên dùng để thu thập dữ liệu từ các tập tin log, có các tính năng thỏa mãn hầu hết các yêu cầu được đặt ra ở trên:
  - Viết bằng Java nên có thể chạy trên mọi hệ điều hành, chỉ cần cài đặt Java Runtime Enviroment.
  - Có khả năng thu thập dữ liệu theo thời gian thực nhờ cơ chế theo dõi sự thay đổi của các tập tin theo thời gian của Logstash.
- Hoạt động của Logstash:

  Quá trình xử lý sự kiện của Logstash trong Log Sensor trải qua hai giai đoạn chính: inputs → outputs. Các input thu thập log, sinh ra sự kiện và các output vận chuyển các sự kiện đến message queue Ngoài ra, các input và output cũng hỗ trợ các codec cho phép encode và decode dữ liệu đầu vào và đầu ra.

  Inputs: sử dụng để đẩy dữ liệu vào Logstash và có hỗ trợ nhiều các cơ chế đọc input như:
  - file: đọc dữ liệu từ các tập tin trên hệ thống tương tự như lệnh tail –f trên Unix
  - syslog: listen trên port 514 của các syslog messages và xử lý theo định dạng RFC3164
  - Microsoft Windows Event Log: đọc dữ liệu trên hệ thống log của Windows
  - TCP/UDP: đọc dữ liệu đến từ luồng TCP/UDP

  Outputs: có tác dụng chuyển dữ liệu log sau khi đã được đọc vào tới hệ thống message queue. Một sự kiện có thể được chuyển tới nhiều đích khác nhau. Một số các output hay được sử dụng là:
  - Kafka: gửi dữ liệu đến Kafka. Đây cũng là output được sử dụng trong Log Sensor.
  - file: ghi dữ liệu ra file trên hệ thống

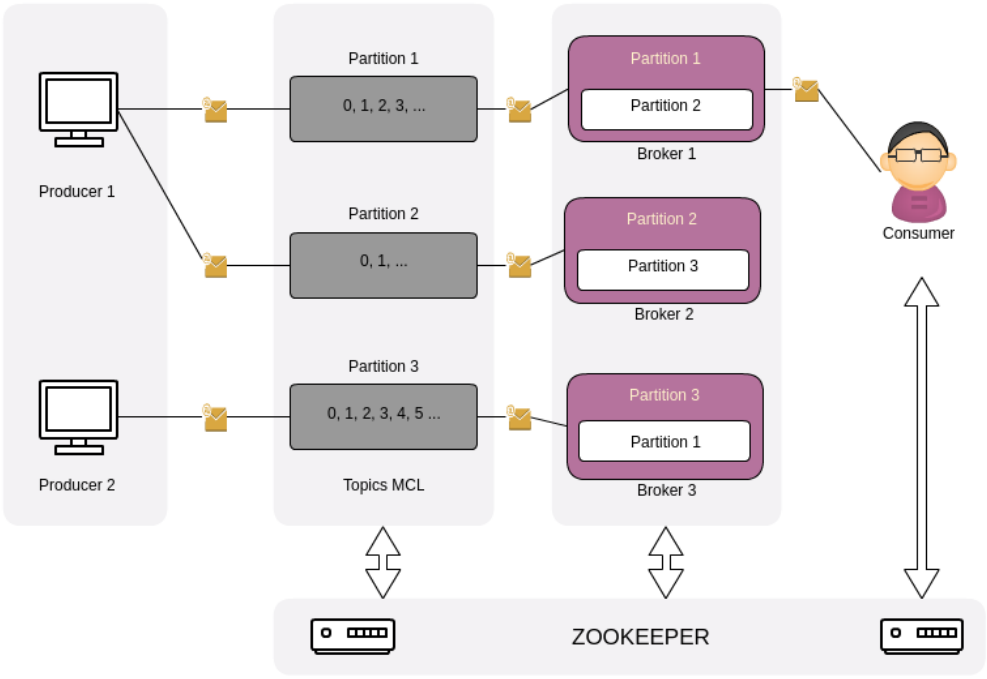
- Elasticsearch: gửi dữ liệu đến Elasticsearch

2.2.2. Message Queue

- Nhiệm vụ: đóng vai trò như một trạm trung chuyển dữ liệu, thu thập dữ liệu này từ các nguồn khác nhau, tạo thành các luồng stream để từ các luồng này có thể chuyển tiếp dữ liệu sang cho các hệ thống khác để xử lý.
- Yêu cầu: Do đóng vai trò là một trạm trung chuyển giữa các hệ thống nên hàng đợi cần đáp ứng một vài tiêu chí sau:
    - Tốc độ nhanh
    - Dễ dàng mở rộng
    - Độ tin cậy cao
    - Có khả năng xử lý trong thời gian thực
- Giải pháp lựa chọn: Apache Kafka. Đây là hệ thống truyền thông điệp phân tán, độ tin cậy cao, dễ dàng mở rộng và có thông lượng cao. Kafka cung cấp cơ chế offset (có thể hiểu như tương tự như chỉ số của một mảng) để lấy thông điệp một cách linh hoạt, cho phép các ứng dụng có thể xử lý lại dữ liệu nếu việc xử lý trước đó bị lỗi. Ngoài ra, cơ chế "đăng kí" theo dõi cho phép việc lấy thông điệp ra gần như tức thời ngay khi dữ liệu đi vào hàng đợi. Kafka được thiết kế hỗ trợ tốt cho việc thu thập dữ liệu theo thời gian thực nên nó có đầy đủ các đặc điểm có thể đáp ứng được những yêu cầu ở phía trên:
    - Tốc độ nhanh: Với một máy đơn cài đặt Kafka có thể xử lý số lượng dữ liệu từ việc đọc và ghi lên tới hàng trăm megabyte trong một giây từ hàng ngàn máy khách.
    - Khả năng mở rộng: Kafka được thiết kế cho phép dễ dàng được mở rộng và trong suốt với người dùng (nghĩa là không có thời gian chết – ngừng hoạt động trong khi thêm một nút máy chủ mới vào cụm). Khi Kafka chạy trên một cụm, luồng dữ liệu sẽ được phân chia và được vận chuyển tới các nút trong cụm, do đó cho phép trung chuyển được các dữ liệu có khối lượng lớn hơn nhiều so với sức chứa của một máy đơn.
    - Độ tin cậy: Dữ liệu vào hàng đợi sẽ được lưu trữ trên ổ đĩa và được sao chép tới các nút khác trong cụm để ngăn ngừa việc mất dữ liệu, như vậy Kafka đảm bảo tính chịu lỗi cao.
- Hoạt động của Kafka:

Kafka sử dụng mô hình truyền thông public-subscribe, bên pulbic dữ liệu được gọi là producer bên subscribe nhận dữ liệu theo topic được gọi là consumer. Kafka có khả năng truyền một lượng lớn dữ liệu trong thời gian thực, trong trường hợp bên nhận chưa nhận dữ liệu vẫn được lưu trữ sao lưu trên một hàng đợi và cả trên ổ đĩa bảo đảm an toàn. Các thành phần chính của Kafka như sau:

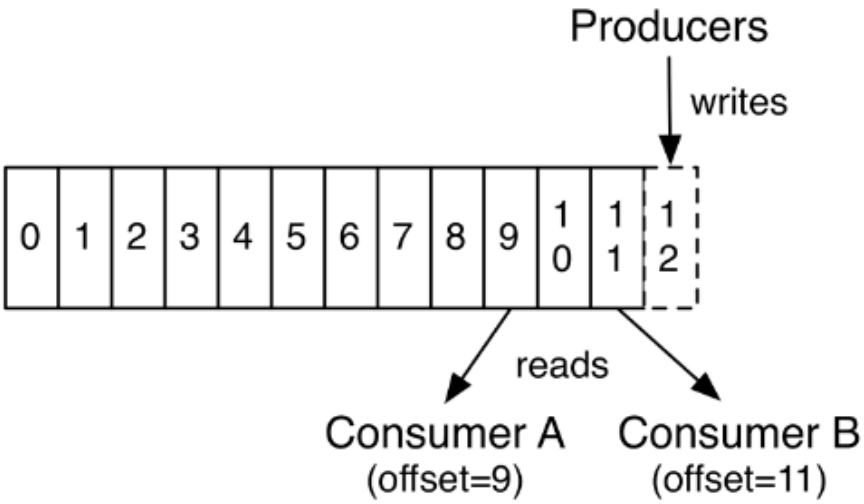
- Topic: là "chủ đề", thông thường các dữ liệu liên quan hoặc tương tự được nhóm vào cùng một chủ đề. Mỗi chủ đề có thể được coi là một nguồn dữ liệu riêng biệt, dữ liệu này được truyền trong kafka theo topic, khi muốn truyền các dữ liệu khác nhau hay truyền dữ liệu cho các ứng dụng khác nhau ta sẽ cần phải tạo ra các topic mới.
- Partition: mỗi topic sẽ được phân chia thành nhiều vách ngăn. Các vách ngăn này là nơi lưu trữ dữ liệu cho 1 topic, trên mỗi các vách ngăn dữ liệu sẽ được lưu theo một thứ tự bất biến và được gán cho một id gọi là offset, được hiểu như chỉ số của một mảng. Offset trên mỗi vách ngăn là độc lập. Một vách ngăn có thể được sao chép trên nhiều máy khác nhau trong một cụm kafka
- Broker: Kafka chạy trên một cụm bao gồm một hoặc nhiều máy (node), mỗi máy được gọi là một broker. Broker là nơi lưu trữ các partition, một broker có thể lưu trữ nhiều partition.
- Producer: sẽ chuyển dữ liệu tới broker. Cụ thể hơn, producer có nhiệm vụ chọn các message giống nhau để đưa vào topic phù hợp, nhiệm vụ này rất quan trọng giúp cho Kafka có khả năng mở rộng tốt.
- Consumer: sẽ đọc dữ liệu từ broker. Kafka là hệ thống sử dụng mô hình truyền thông public-subscribe nên mỗi một topic có thể đc xử lý bởi nhiều consumer khác nhau, miễn là consumer đấy subcribe topic đấy.



Hình 2. 2: Sơ đồ hệ thống truyền dữ liệu trên Kafka

a. Topic

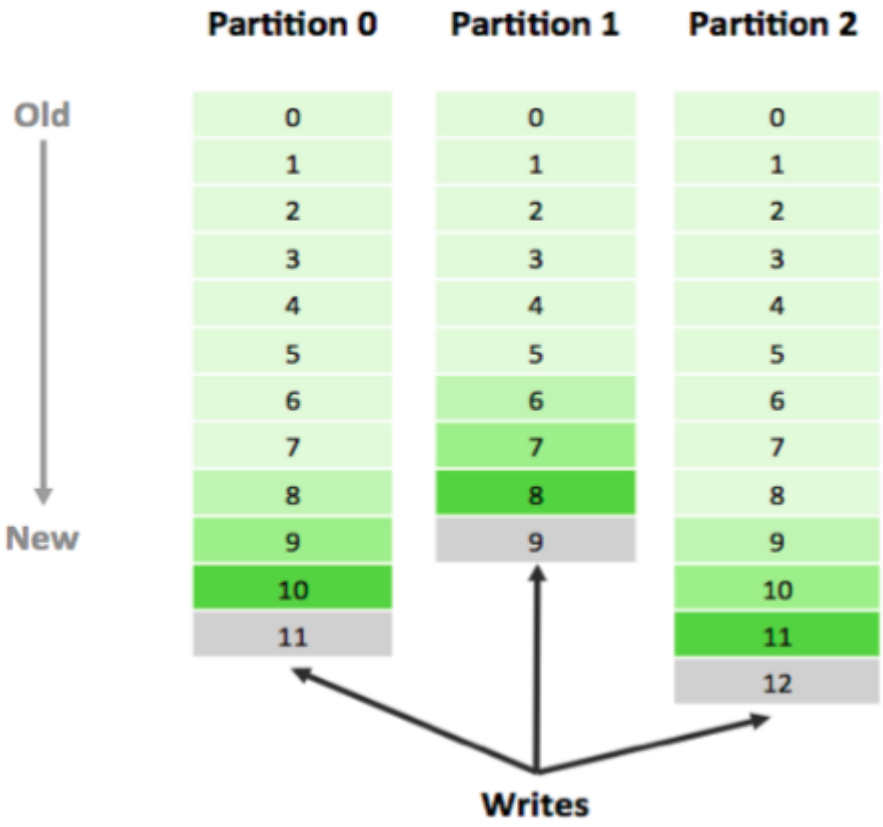
Mỗi topic sẽ có tên do người dùng đặt và có thể coi mỗi topic như là một hàng đợi của thông điệp. Các thông điệp mới do một hoặc nhiều producer đẩy vào, sẽ luôn luôn được thêm vào cuối hàng đợi. Bởi vì mỗi thông điệp được đẩy vào topic sẽ được gán một offset tương ứng nên consumer có thể dùng offset này để điều khiển quá trình đọc thông điệp.



Hình 2. 3: Mô hình hoạt động của Topic

b. Partition

Với mỗi partition, tùy thuộc vào người dùng cấu hình sẽ có một số bản sao chép nhất định để đảm bảo dữ liệu không bị mất khi một node trong cụm bị hỏng, tuy nhiên số lượng bản sao không được vượt quá số lượng broker trong cụm, và những bản sao đó sẽ được lưu lên các broker khác. Broker chứa bản chính của partition gọi là broker "leader". Những bản sao chép này có tác dụng giúp hệ thống không bị mất dữ liệu nếu có một số broker bị lỗi, với điều kiện số lượng broker bị lỗi không lớn hơn hoặc bằng số lượng bản sao của mỗi partition.



Hình 2. 4: Mô hình hoạt động của Partition

c. Producer

Producer có nhiệm vụ đẩy dữ liệu vào một hoặc nhiều topic. Người dùng có thể quyết định liệu những thông điệp nào sẽ cùng thuộc vào một partition thông qua một chuỗi khóa đính kèm với thông điệp. Nếu không producer sẽ gán một khóa ngẫu nhiên và quyết định đích đến của thông điệp dựa trên giá trị băm của khóa.

d. Consumer

Consumer có nhiệm vụ kéo dữ liệu từ một topic chỉ định về. Tùy thuộc vào mục đích sử dụng, Kafka cung cấp hai hàm API như sau:

- High Level Consumer: API này hướng tới những ứng dụng không quan tâm về việc điều khiển việc đọc thông điệp, người dùng chỉ có thể đọc từ thông điệp cũ nhất hoặc đọc từ thông điệp mới nhất. API này luôn lưu lại offset của thông điệp được lấy về mới nhất của mỗi partition vào Zookeeper.
- Simple Consumer: Việc sử dụng API này tương đối phức tạp hơn API trên nhưng nó cho phép điều khiển việc đọc một cách linh hoạt dựa trên offset. Do đó, API này cho phép ứng dụng có thể xử lý lại thông điệp nếu gặp lỗi trong quá trình xử lý trước đó.

e. Zookeeper

Kafka sẽ có những Broker để điều phối các streaming của dữ liệu, ngoài ra Kafka sử dụng Zookeeper để:

- Quản lý, điều phối các Broker với nhau.
- Quản lý sự trao đổi giữa các Broker với các Consumer.

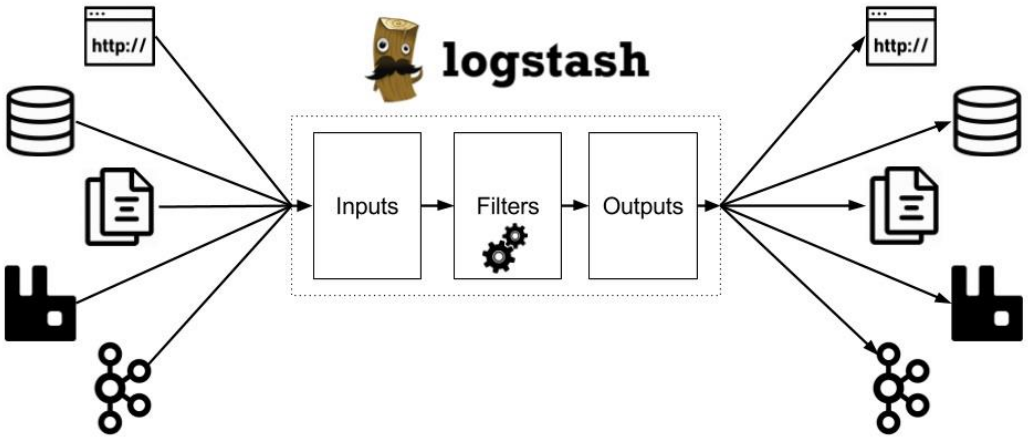
2.2.3. Log Collector

- Nhiệm vụ: đọc các thông tin theo từng topic từ máy chủ Kafka và chuẩn hóa các thông tin nhận được dựa trên những quy tắc chuẩn hóa do người quản trị tự cấu hình.
- Yêu cầu:
  - Có độ tương thích cao: Cần phải đọc và tiếp nhận được toàn bộ các thông tin được chuyển đến từ hệ thống Kafka
  - Thời gian thực: Log Collector phải đáp ứng được yêu cầu là luôn luôn sẵn sàng khả dụng để có thể lắng nghe được các thông tin từ bên máy chủ Kafka gửi sang để có thể đưa ra được kết quả phân tích và đánh giá theo thời gian thực.
- Giải pháp lựa chọn: Logstash. Đây là phần mềm mã nguồn mở có chức năng chính trong việc thu thập và phân tích dữ liệu từ các tập tin log. Có đầy đủ các tính năng thỏa mãn những yêu cầu đã được đặt ra ở trên:
  - Có khả năng thu thập dữ liệu theo thời gian thực bằng các plugin có sẵn ở trong Logstash để luôn luôn lắng nghe được các dữ liệu gửi tới từ Kafka
  - Hỗ trợ viết thêm các luật trong quá trình thu thập log: điều này rất quan trọng vì mỗi loại log lại có một kiểu cấu trúc khác nhau nên việc hỗ trợ việc

viết thêm các luật để phân tích log sẽ giúp Logstash có thể đọc được nhiều các kiểu log khác nhau

- Hoạt động của Logstash:

Toàn bộ quá trình xử lý sự kiện của Logstash trong Log Collector trải qua ba giai đoạn chính là: inputs → filters → outputs. Các input thu thập, tiếp nhận thông tin từ máy chủ Kafka sau đó các thông tin này sẽ được chuyển tới giai đoạn filter để chỉnh sửa, phân tích để đưa ra được các kết quả như yêu cầu đề ra của người quản trị và các kết quả sau khi được phân tích này sẽ được đẩy tới output để vận chuyển các kết quả này đến Elasticsearch. Log Collector sẽ có thêm một giai đoạn khác với Log Sensor, tuy nhiên lại vô cùng quan trọng đó là giai đoạn filter. Luồng xử lý sự kiện của Logstash trong Log Collector có cấu trúc như hình 2.2.



Hình 2. 5: Mô hình hoạt động của Partition

Filters: Giai đoạn filter cho phép người dùng chỉnh sửa, thay đổi, trích xuất một phần dữ liệu của các input đầu vào. Có thể kết hợp sử dụng nhiều filter cùng lúc để có thể đưa ra được kết quả tối ưu nhất. Một số các filter được áp dụng trong Log Collector là:

- grok: giúp phân tích và cấu trúc hóa dữ liệu văn bản bất kỳ. Hiện tại grok là cách tốt nhất trong Logstash để xử lý các dữ liệu thô, biến chúng thành dữ liệu có cấu trúc và có thể tìm kiếm được. Grok cũng có sẵn các mẫu để áp dụng cho một số loại log phổ biến.
- mutate: thực hiện thay đổi, chỉnh sửa trên dữ liệu
- drop: bỏ qua hoàn toàn dữ liệu
- clone: copy dữ liệu, có thể thêm hoặc xóa bớt các trường
- geoip: thêm những thông tin về vị trí địa lý của địa các địa chỉ IP

2.2.4. Database

- Nhiệm vụ: Lưu trữ và đánh index cho các dữ liệu log thu thập được để phục vụ cho việc tìm kiếm sau này
- Yêu cầu: Do các đầu vào dữ liệu thu thập được thường sẽ là rất lớn nên Databse cần đáp ứng các tiêu chí sau:
    - Khả năng tìm kiếm nhanh trên nguồn dữ liệu lớn.
    - Khả năng mở rộng (scalability).
    - Phân tích theo thời gian thực
- Giải pháp lựa chọn: Elasticsearch. Đây cũng là một phần mềm mã nguồn mở, một máy chủ tìm kiếm được xây dựng trên Lucene và là một thư viện full text search khá phổ biến. Elasticsearch hoàn toàn đáp ứng được đầy đủ các yêu cầu nêu ở trên:
    - Toàn bộ dữ liệu trong Elasticsearch đều được index, quá trình tìm kiếm là thực hiện trên dữ liệu đã index.
    - Là Near Real-time Platform: có độ trễ rất nhỏ từ lúc dữ liệu được index tới khi nó có thể tìm kiếm được.
    - Hỗ trợ Cluster, Node, Shard, Replication,… giúp lưu trữ và tìm kiếm dữ liệu phân tán. Đáp ứng khả năng mở rộng theo cả chiều ngang lẫn chiều dọc.
    - Với khả năng thực hiện trích xuất dữ liệu siêu nhanh từ hầu như tất cả các nguồn dữ liệu có cấu trúc hoặc không có cấu trúc. Elasticsearch là công cụ đáp ứng được các vấn đề về hiệu năng cũng như vấn đề về tốc độ trong việc xử lý các thông tin theo thời gian thực.
- Hoạt động của Elasticsearch:

  Đơn vị dữ liệu nhỏ nhất trong Elasticsearch là một field, thuộc một kiểu dữ liệu định sẵn. Một field có một giá trị đơn, như kiểu số hay kiểu string, hoặc là một danh sách các giá trị thuộc cùng một kiểu, như mảng.

  Document là một tập hợp bao gồm các field, tạo thành đơn vị cơ bản được lưu trữ trong Elasticsearch, tương đương với một dòng trong cơ sở dữ liệu quan hệ. Document được coi là đơn vị dữ liệu cơ bản của Elasticsearch là do mọi thao tác cập nhật trên field cũng sẽ cập nhật hoàn toàn một document. Elasticsearch là một hệ thống hướng document: không chỉ lưu trữ chúng, Elasticsearch index toàn bộ nội dung của document theo thứ tự để chúng có thể được tìm kiếm. Elastichsearch sử dụng JSON làm định dạng dữ liệu chính. JSON được hỗ trợ bởi hầu hết các

ngôn ngữ lập trình và đã trở thành định dạng chuẩn cho cơ sở dữ liệu NoSQL bởi tính đơn giản, ngắn gọn và dễ sử dụng. Một document thể hiện thông tin của một cá nhân được mô tả như sau:

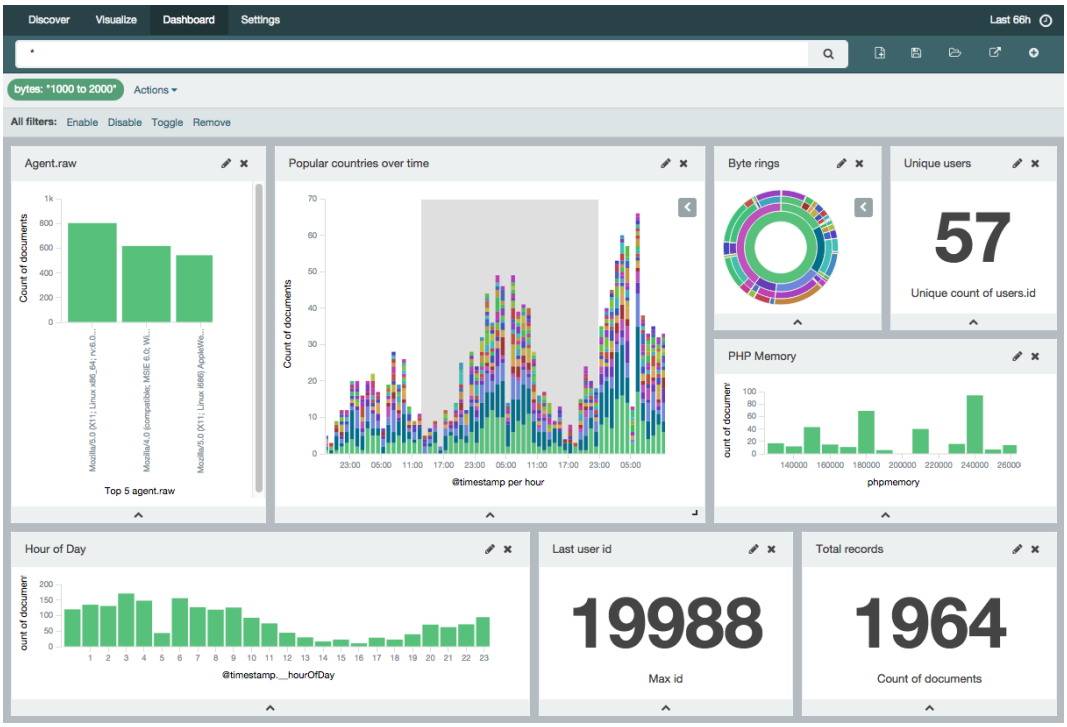
```
{
    "_id": 1,
    "email":      "john@smith.com",
    "first_name": "John",
    "last_name":  "Smith",
    "info": {
        "bio":         "Eco-warrior and defender of the weak",
        "age":         25,
        "interests": [ "dolphins", "whales" ]
    },
    "join_date": "2014/05/01"
}
```

Hình 2. 6: Thông tin chi tiết Elasticsearch

2.2.5. Analysis tool

- Nhiệm vụ: tính toán để tạo ra bản báo cáo số liệu thống kê dựa trên các dữ liệu được đọc ở trong Elasticsearch và hiển thị kết quả tính toán lên trình duyệt web
- Yêu cầu:
    - Tương thích: Yêu cầu phải kết nối được và truy cập được các dữ liệu được lưu ở trong Elasticsearch
    - Chính xác: Cần phải hiển thị đúng được các kết quả lưu trong Elasticsearch cũng như đưa ra được các thông số tính toán chính xác để phục vụ cho việc theo dõi của người quản trị hệ thống
- Giải pháp lựa chọn: Kibana. Đây là một công cụ phân tích mã nguồn mở, trực quan được thiết kế để làm việc với Elasticsearch. Kibana có giao diện web có thể được sử dụng để tìm kiếm, xem và tương tác với dữ liệu được lưu trữ tại các bản ghi mà Logstash đã lập chỉ mục. Kibana đáp ứng được yêu cầu trên dựa vào các tính năng:
    - Cho phép bạn tạo và chia sẻ nhanh chóng các biểu đồ động hiển thị những thay đổi để Elasticsearch truy vấn trong thời gian thực.
    - Có thể dễ dàng thực hiện phân tích dữ liệu tiên tiến và hiển thị dữ liệu của bạn trong một loạt các biểu đồ, bảng biểu và bản đồ.
    - Tận dụng được khả năng tìm kiếm và index mạnh mẽ của Elasticsearch để hiển thị các giao diện cho người dùng.
- Hoạt động của Kibana:

Kibana sẽ đọc các dữ liệu, thông tin đã được lưu trữ sẵn ở trong Elasticsearch. Các dữ liệu này sẽ phục vụ cho việc tính toán để tạo ra các biểu đồ, hình ảnh chi tiết và toàn bộ các thông tin trên đều được hiển thị theo thời gian thực để có thể đưa ra được những số liệu, phân tích cụ thể giúp người quản trị dễ dạng đưa ra các kết luận đối với các hệ thống được giám sát. Các giao diện đồ họa của Kibana sẽ được hiển thị hoàn toàn trên nền web. Các biểu đồ thông số cơ bản được hiển thị trên Kibana có cấu trúc như hình 2.4:



Hình 2. 7: Các biểu đồ mẫu trên Kibana

2.3. Phi chức năng

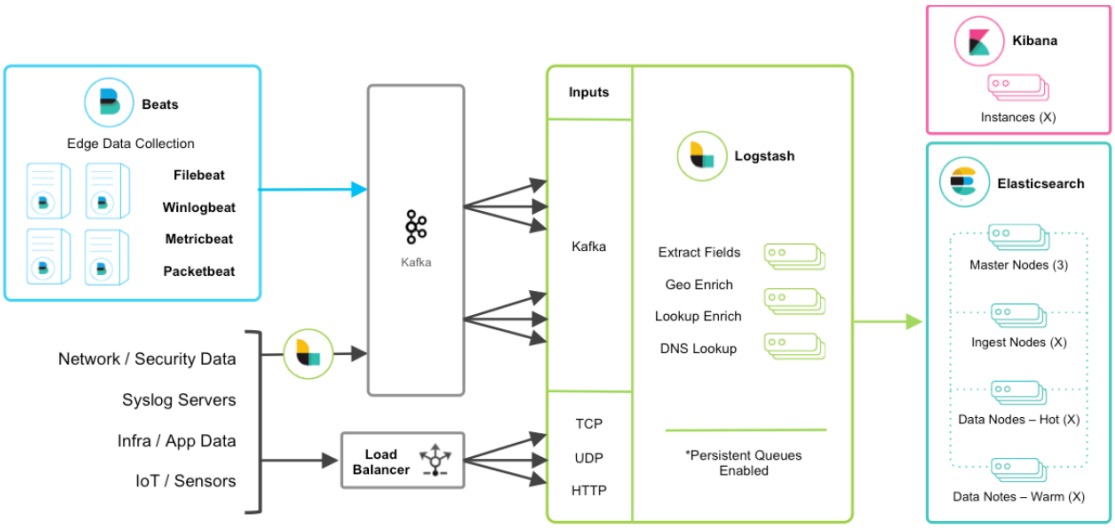
2.3.1. Các vấn đề về bảo mật

- Bào mật đường truyền: Mã hóa đường truyền sẽ được áp dụng cho cả 2 quá trình truyền dữ liệu từ Log Sensor đến Kafka và quá trình nhận dữ liệu từ Kafka đến Log Collector. Cả hai quá trình này đều sẽ được áp dụng kĩ thuật bảo mật SSL/TLS để có thể bảo vệ thông tin trong các giai đoạn truyền và nhận dữ liệu
- Bảo vệ Elasticsearch: Ở đây sẽ sử dụng phần mềm X-pack. Với công cụ mã nguồn mở X-pack sẽ cho phép chúng ta bảo vệ được các thành phần trong toàn bộ hệ thống. Hệ thống khi cài X-pack sẽ được bảo vệ bằng mật khẩu cũng như thực hiện các biện pháp bảo mật tiên tiến hơn như mã hóa đường truyền, kiểm soát truy cập dựa trên vai trò và lọc IP.

- Ngăn chặn truy cập trái phép: Để ngăn chặn các truy cập trái phép vào cụm tìm kiếm Elasticsearch, hệ thống cần phải có các biện pháp cụ thể để có thể xác thực được định danh của những người sử dụng. Điều này đơn giản có nghĩa là chúng ta cần một cách để xác nhận rằng người dùng có quyền truy cập vào hệ thống hay không. Trong nhiều trường hợp, chỉ đơn giản xác thực được định danh của người dùng không là chưa đủ. Hệ thống cần phải có một cách khác để có thể kiểm soát được các thông tin của người dùng như người dùng có quyền truy cập vào hệ thống hay không? nhiệm vụ của họ ở trong hệ thống là gì? Họ có quyền truy cập đến các tài nguyên hạn chế hay không. X-Pack sẽ cho phép hệ thống ủy quyền cho người sử dụng bằng cách gán truy câp đặc quyền đối với từng vai trò và gán các vai trò đó cho người sử dụng. Bảo mật X-Pack cũng hỗ trợ ủy quyền dựa trên IP, hệ thống có thể đưa vào danh sách trắng và danh sách đen các địa chỉ IP hoặc mạng con cụ thể để kiểm soát được quyền truy cập của từng người dụng vào trong các máy chủ.
- Bảo toàn tính toàn vẹn của dữ liệu: Một phần quan trọng của an toàn thông tin là giữ bí mật dữ liệu. X-Pack bảo toàn tính toàn vẹn dữ liệu của hệ thống bằng cách mã hóa các thông tin liên lạc đến và đi từ các nút dựa trên công nghệ mã hóa SSL/TLS. Để muôn hệ thống có thể được bảo vệ tốt hơn nữa, có thể tăng cường độ mã hoá trong từng phần của các nút.

2.3.2. Tính khả mở của hệ thống

Một hệ thống quản lý an ninh tốt phải đưa ra được khả năng quản trị trong thực tế tốt, có những khả năng mở rộng và phát triển dễ dàng trong tương lai. Dưới đây là một mô hình đề xuất về hướng mở rộng của hệ thống trong tương lai:



Hình 2. 8: Hướng mở rộng của hệ thống

- Việc mở rộng hệ thống trước tiên cần phải tập trung đến vấn đề bổ sung thêm các nguồn đầu vào cho hệ thống, có các cơ chế khác nhau để thu thập dữ liệu, dễ dàng tích hợp và tập trung chúng vào ELK Stack. Như hình 2.8 hệ thống sẽ được bổ sung thêm các nguồn đầu vào từ App Data, Network, Infrastructure hoặc là từ các thiết bị IoT. Hệ thống cần bổ sung thêm các cơ chế thu thập dữ liệu từ các ứng dụng mã nguồn mở khác nhau như Beats. Ngoài ra cũng có thể kể đến một vài các cơ chế thu thập dữ liệu khác như:
  - Các giao thức TCP, UDP và HTTP là những cách phổ biến để đưa dữ liệu vào Logstash. Logstash có thể tạo ra các node tương ứng để lắng nghe với các plugin đầu vào như TCP, UDP và HTTP.
  - Các công nghệ máy chủ syslog hiện có như rsyslog và syslog-ng thường gửi syslog đến các thiết bị đầu cuối Logstash qua các giao thức như TCP hoặc UDP để xử lý. Nếu định dạng dữ liệu này phù hợp với RFC3164, nó có thể được đưa trực tiếp vào trong Logstash .
- Tiếp theo của việc mở rộng hệ thống sẽ cần quan tâm đến vấn đề cân bằng tải và bộ nhớ đệm cho hệ thống. Đó chính là phần Load Balancer như ở trên hình 2.8. Dưới đây là một số các hướng mở rộng cho phần cân bằng tải của hệ thống:
  - Đối với các hệ thống lớn thì cần thêm phần cân bằng tải bằng các phần mềm bên thứ ba như HAProxy hoặc các công nghệ hàng đợi khác như RabbitMQ, ActiveMQ, ZeroMQ hoặc Redis để có thể cân bằng số lưu lượng truy cập đến một nhóm các nút Logstash.
  - Ngoài ra hệ thống cũng có thể bổ sung thêm nhiều các node Kafka để có thể đạt được tính sẵn sàng và hiệu năng cao hơn.

- Cuối cùng hệ thống có thể mở rộng thêm bằng cách bổ sung các node cho phần xử lý, như trên hình 2.8 hệ thống đã được bổ sung thêm các node cho Logstash, Elasticsearch và Kibana để có thể đạt được hiệu năng và tính sẵn sàng cao nhất. Dưới đây sẽ trình bày chi tiết việc bổ sung các node cho hệ thống xử lý:

  - Logstash khi triển khai, bản thân nó đã được coi là 1 cluster với 1 node duy nhất. Chính vì vậy khi mà có nhiều các dữ liệu đầu vào truyền đến logstash thì sẽ dẫn đến tình trạng quá tải và làm không xử lý kịp dữ liệu. Để đảm bảo việc xử lý dữ liệu diễn ra theo thời gian thực thì chúng ra cần tối thiểu 2 node logstash, trong đó 1 node sẽ tiếp nhận luồng dữ liệu từ hàng đợi Kafka và 1 node sẽ tiếp nhận luồng dữ liệu từ các giao thức căn bản như TCP, UDP, HTTP…..

  - Elasticsearch khi triển khai, bản thân nó đã được coi là 1 cluster với 1 node duy nhất, nó tạo ra 5 shards để chứa dữ liệu của bạn. Để đảm bảo dữ liệu được an toàn, không bị mất mát, elasticsearch muốn nhân bản - replicas mỗi 1 shards ra một bản sao khác. Khi này 5 shards ban đầu là không đủ, cần phải thêm một vài các node elasticsearch mới(tối thiểu là 2 node elasticsearch) để tạo ra các shards phục vụ cho việc lưu trữ các bản sao. Từ các lần sau, elasticsearch khi truy vấn sẽ sử dụng replicas lưu trữ ở node 2 hoặc primary shard lưu trữ ở node 1. Điều này làm tăng performance cho hệ thống

2.4. Kết chương

Trong chương 2 đã đưa ra được các giải pháp tương ứng cho từng thành phần của một hệ thống giám sát an toàn mạng. Đưa ra được mô hình tổng quan và cách thức hoạt động của các thành phần được lựa chọn. Trình bày được cách thành phần phi chức năng của cả hệ thống

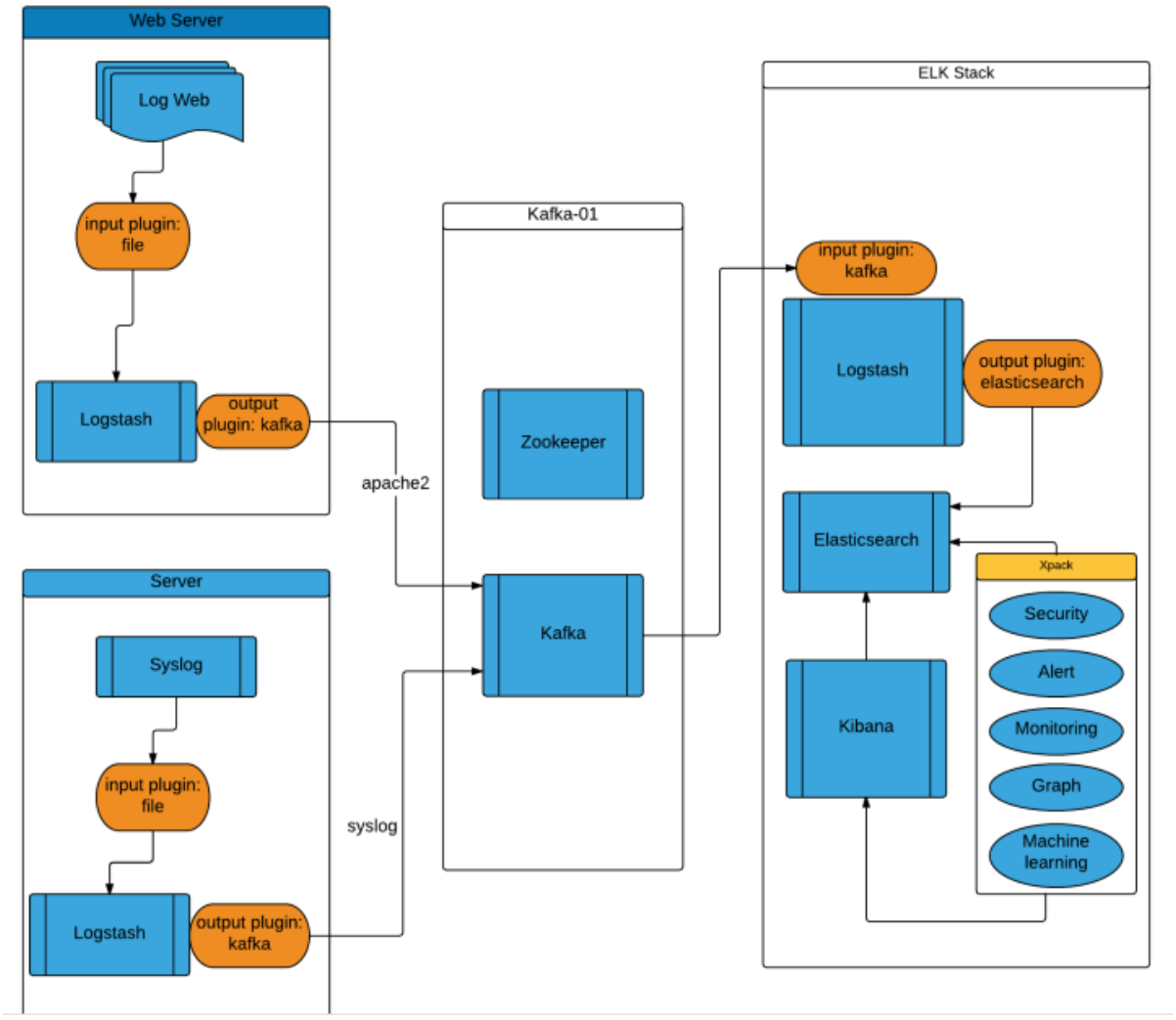
CHƯƠNG 3. TRIỂN KHAI HỆ THỐNG PHẦN MỀM

Chương này trình bày về các vấn đề sau:

- Đưa ra được mô hình triển khai
- Cấu hình và triển khai hệ thống
- Thử nghiệm hệ thống và một số kết quả thu được

3.1. Mô hình triển khai

Từ mô hình chung đã được đề xuất ở chương 2 cho việc xây dựng mô hình giám sát an toàn mạng. Ở chương này em xin trình bày mô hình triển khai cụ thể cho từng thành phần. Môi trường triển khai sẽ ở trên các máy ảo được tạo bởi phần mềm Vmware, trong đó sẽ có 3 máy ảo tương ứng với 3 phần chính là Log Sensor bao gồm Web server và Server, Message Queue bao gồm Kafka-01 và Log Colector bao gồm ELK Stack. Mô hình triển khai cụ thể như hình 3.1:



Hình 3. 1: Mô hình triển khai của hệ thống

- Log Sensor

  Cấu hình:

  - Phần mềm sử dụng:

    Logstash

    - Version: 6.0.1
    - Release date: December 06, 2017

  - Nền tảng triển khai:
    - Hệ điều hành Ubuntu 14.04
    - Ram: >= 2GB
    - CPU: >= 2

  Chức năng:

  - Web server: Là một máy chủ web và được cài đặt sẵn Logstash có nhiệm vụ chính là tương tác với các tập tin log của web server để nhận biết trạng thái của những file log đó. Sau đó sẽ chuyển các thông tin của tập tin log này sang máy chủ Kafka.
  - Server: Là một máy chủ được cài đặt sẵn Logstash có nhiệm vụ chính là tương tác với các tập tin syslog để nhận biết trạng thái của những tập tin log đó. Sau đó sẽ chuyển các thông tin của tập tin log này sang máy chủ Kafka.

- Kafka

  Cấu hình:

  - Phần mềm sử dụng:

    Kafka

    - Version: 2.11-1.0.0
    - Release date: November 1, 2017

  - Nền tảng triển khai:
    - Hệ điều hành Ubuntu 14.04
    - Ram: >= 2GB
    - CPU: >= 2

- Chức năng:
  - Có nhiệm vụ chính là thu thập dữ liệu này 2 máy chủ là Web server và Server, để tạo thành 2 luồng stream dữ liệu là Log web và Syslog sau đó từ

2 luồng stream này sẽ cung cấp lại thông tin cho hệ thống Log Collector để xử lý từng luồng dữ liệu khác nhau.

- Log Collector

  Cấu hình:

  - Phần mềm sử dụng:

    Logstash

    - Version: 6.0.1
    - Release date: November 1, 2017

    Elasticsearch

    - Version: 6.0.1
    - Release date: November 1, 2017

    Kibana

    - Version: 6.0.1
    - Release date: November 1, 2017

  - Nền tảng triển khai:
    - Hệ điều hành Ubuntu 14.04
    - Ram: >= 2GB
    - CPU: >= 2

- Chức năng:
  - Logstash: Có nhiệm vụ chính là đọc những thông tin thu được từ hai luồng dữ liệu là Log web và Syslog được chuyển trực tiếp từ máy chủ Kafka, sau đó sẽ chuẩn hóa thông tin vừa nhận được dựa trên những quy tắc những rule đã viết từ trước.
  - Elasticsearch: Ở đây sau khi Logstash đã phân tích xong 2 luồng dữ liệu là Log web và Syslog sẽ đẩy xuống cho Elasticsearch và tại đây Elasticsearch có nhiệm vụ chính là một công cụ lưu trữ tất cả các thông tin vừa nhận được từ Logstash.
  - Kibana: Sau khi đã lưu trữ tất cả dữ liệu vào trong Elasticsearch. Sau đó Kibana có nhiệm vụ chính là đọc các dữ liệu dạng ở trong Elasticsearch dưới dạng JSON từ 2 nguồn dữ liệu là Log web và Syslog, sau đó sẽ tính toán để tạo ra các biểu đồ, bảng biểu và các báo cáo số liệu thống kê dựa trên những thông số nhận được từ 2 nguồn dữ liệu

- Xpack: là một phần mở rộng của hệ thống cung cấp các chức năng an ninh về cảnh báo, theo dõi, báo cáo và tạo biểu.

3.2. Các phần mềm hỗ trợ khác

3.2.1. *JDK - Java Development Kit.*

Bộ công cụ cho người phát triển ứng dụng bằng ngôn ngữ lập trình Java.

- Tại sao cần phải cài JDK? Vì Logstash và Elasticsearch yêu cầu Java nên cần có một Java Virtual Machine để hoạt động. Vì vậy trước khi cài đặt những phần mềm phía trên hệ thống cần phải cài đặt JDK cho máy chủ.
- Phiên bản sử dụng: JDK 8

3.2.2. *Nginx*.

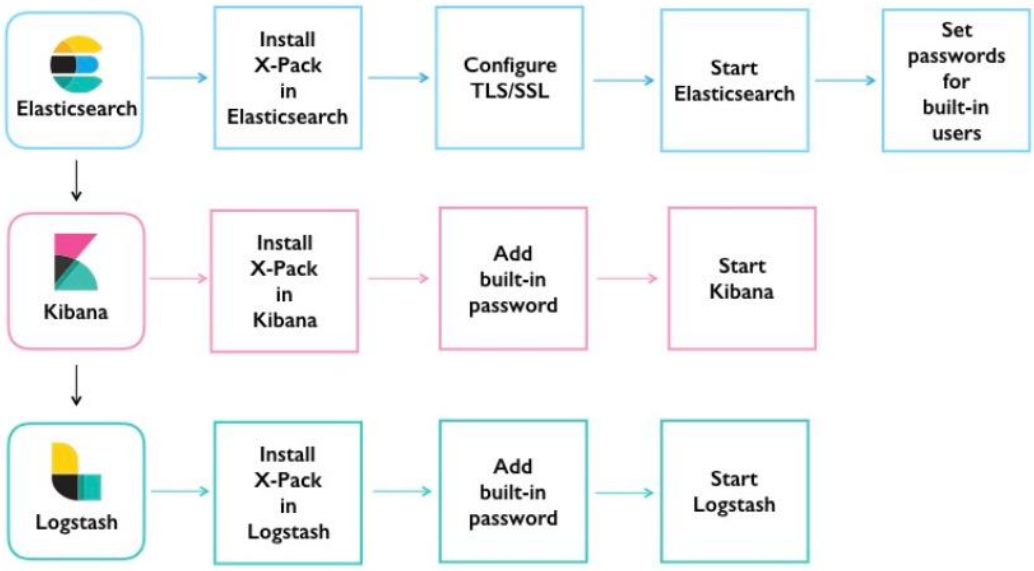
Là một máy chủ proxy ngược mã nguồn mở (open source reverse proxy server) sử dụng phổ biến giao thức HTTP, HTTPS, SMTP, POP3 và IMAP , cũng như dùng làm cân bằng tải (load balancer), HTTP cache và máy chủ web (web server).

- Tại sao cần Nginx? Vì Kibana cũng là một máy chủ web. Vì vậy nên hệ thống cần phải cài đặt thêm nginx để giảm thiểu đi các vấn đề mà một máy chủ web thường gặp phải là vấn đề bảo mật và cân bằng tải.
- Phiên bản sử dụng: nginx-1.13.7

3.2.3. Xpack

Là một phần mở rộng Elastic Stack cung cấp các chức năng an ninh về cảnh báo, theo dõi, báo cáo và biểu đồ thành một gói để có thể dễ dàng cài đặt.

- Tại sao cần Xpack? các thành phần cài đặt X-Pack có thể làm việc cùng nhau một cách liền mạch, và có thể dễ dàng kích hoạt hoặc vô hiệu hóa các tính năng mà mình muốn sử dụng. Xpack còn có một vài các tính năng khác như là về Xpack-Graph để có thể tạo các bảng biểu đối với các dữ liệu trông chuyên nghiệp hơn. Thiết lập Xpack-Machine learning để có thể tự động phát hiện những thay đổi bất thường trong dữ liệu của bạn. Xpack-Monitoring và Xpack-Report định dạng các cấu hình cảnh báo để kích hoạt thông báo về thay đổi hoặc lên lịch, gửi báo cáo định kỳ hoặc đưa ra màn hình để có thể quản lý được toàn bộ hệ thống của mình. Ngoài ra còn có Xpack-Security để đảm bảo về vấn đề an toàn bảo mật cho hệ thống
- Phiên bản sử dụng: version 6.0.0
- Mô hình hoạt động của Xpack trong Log Collector



Hình 3. 2: Sơ đồ thông tin chi tiết của Xpack

3.3. Cấu hình hệ thống

3.3.1. Cài đặt Java 8

Sử dụng các lệnh phía dưới để cài đặt Java 8 cho máy chủ:

```
#sudo add-apt-repository –y ppa:webupd8team/java
#sudo apt-get update
#sudo apt-get –y install oracle-java8-installer

NOTE:Trong quá trình cài sẽ hỏi về license, chọn accept để hoàn thành
```

Sau khi cài đặt hoàn thành có thể sử dụng lệnh *#java –version* để kiểm tra xem đã cài đặt JDK 8 thành công hay chưa. Kết quả cài đặt thành công được hiển thị như hình 3.3:

```
quanghung@ubuntu-server:~$ java -version
java version "1.8.0_151"
Java(TM) SE Runtime Environment (build 1.8.0_151-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.151-b12, mixed mode)
quanghung@ubuntu-server:~$
```

Hình 3. 3: Kết quả kiểm tra sau khi cài JDK

3.3.2. Cài đặt Logstash

Có thể sử dụng các dòng lệnh phía dưới để cài đặt Logstash cho máy chủ:

```
#sudo wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add –
#sudo apt-get install apt-transport-https
#echo "deb https://artifacts.elastic.co/packages/6.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-6.x.list
#sudo apt-get update
#sudo apt-get install logstash
```

3.3.3. Cài đặt Elasticsearch

Có thể sử dụng các dòng lệnh phía dưới để cài đặt Elasticsearch cho máy chủ:

```
#sudo wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add –
#sudo apt-get install apt-transport-https
#echo "deb https://artifacts.elastic.co/packages/6.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-6.x.list
#sudo apt-get update
#sudo apt-get install elasticsearch
```

Sau khi Elasticsearch cài đặt xong, thực hiện chỉnh sửa file cấu hình ở /etc/elasticsearch/elasticsearch.yml để hạn chế người ngoài truy cập vào Elasticsearch (port 9200) để đọc dữ liệu hoặc tắt cụm ElasticSearch thông qua các API HTTP. Ta tìm đến dòng *network.host*, bỏ ghi chú và thay thế giá trị của nó với *localhost.*

```
#network.host: localhost

#cluster.name: elasticsearch
```

Lưu file cấu hình, khởi động lại Elasticsearch và thiết lập elasticsearch khởi động cùng hệ điều hành:

```
#sudo /etc/init.d/elasticsearch restart

#sudo update-rc.d elasticsearch defaults
```

Sử dụng lệnh: curl http://localhost:9200 --user elastic:123456 để kiểm tra xem máy chủ elasticsearch đã hoạt động chưa. Kết quả cài đặt elasticsearch thành công như hình 3.4:

```
quanghung@ubuntu-server:~$ curl http://localhost:9200 --user elastic:123456
{
  "name" : "lCz_RR3",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "uzt5Y-j7SYSH-Q8gdw4W2w",
  "version" : {
    "number" : "6.0.0",
    "build_hash" : "8f0685b",
    "build_date" : "2017-11-10T18:41:22.859Z",
    "build_snapshot" : false,
    "lucene_version" : "7.0.1",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Hình 3. 4: Kết quả kiểm tra sau khi cài Elasticsearch

3.3.4. Cài đặt Kibana

Có thể sử dụng các dòng lệnh phía dưới để cài đặt Kibana cho máy chủ:

```
#sudo wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo
apt-key add –
#sudo apt-get install apt-transport-https
#echo "deb https://artifacts.elastic.co/packages/6.x/apt stable main" | sudo
tee -a /etc/apt/sources.list.d/elastic-6.x.list
#sudo apt-get update
#sudo apt-get install kibana
```

Lưu file cấu hình, khởi động lại Kibana và thiết lập Kibana khởi động cùng hệ điều hành:

```
#sudo /etc/init.d/kibana restart
#update-rc.d kibana defaults
```

3.3.5. Cài đặt Nginx

Có thể sử dụng các dòng lệnh phía dưới để cài đặt Nginx cho máy chủ:

#sudo apt-get -y install nginx

Sử dụng openssl tạo user admin cho phép truy cập vào giao diện web của Kibana

#echo "kibanaadmin:`openssl passwd -apr1`" | sudo tee –a /etc/nginx/htpasswd.users

Sử dụng lệnh trên sẽ tạo user *kibanaadmin*, password sẽ là pass mà bạn nhập sau khi chạy lệnh trên.

Thực hiện cấu hình Nginx theo file sau để cho phép nginx truy cập vào Kibana. Chỉnh sửa lại file cấu hình như sau:

```
server {
        listen 80 default_server;
        listen [::]:80 default_server ipv6only=on;

        root /usr/share/nginx/html;
        index index.html index.htm;

        # Make site accessible from http://localhost/
        server_name 192.168.196.134;

        auth_basic "Restricted Access";
        auth_basic_user_file /etc/nginx/htpasswd.users;


        location / {
                proxy_pass http://localhost:5601;
                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection 'upgrade';
                proxy_set_header Host $host;
                proxy_cache_bypass $http_upgrade;
        }
}
```

Hình 3. 5: File cấu hình nginx

File cấu hình Nginx trên sẽ chuyển tất cả các lưu lượng HTTP vào Kibana, Kibana listen tại *localhost:5601*. Nginx sẽ sử dụng htpasswd.users tạo phía trên làm yêu cầu xác thực. Kiểm tra cấu hình Nginx và khởi động lại:

```
#/etc/init.d/nginx restart
```

NOTE: Nếu server sử dụng firewall thì phải mở port cho phép truy cập từ internet.

```
# ufw allow 'Nginx Full'

# ufw reload
```

Hiện tại đã có thể truy cập vào Kibana thông qua Nginx bằng địa chỉ: http://ip_public_elk_server bằng user kibanaadmin, mật khẩu được nhập ở bước trên.

3.3.6. Cài đặt Kafka

Có thể sử dụng các dòng lệnh phía dưới để cài đặt Kafka cho máy chủ:

```
#wget http://apache.mirror.anlx.net/kafka/0.8.2.0/kafka_2.10-0.8.2.0.tgz
#tar -xzf kafka_2.10-0.8.2.0.tgz
```

Bộ cài kafka đi kèm với ZooKeeper server rồi nên nếu bạn chưa có zookeeper ở local thì bạn có thể start một server Zookeeper đi kèm bộ cài, chạy dưới dạng một single node:

```
#sudo bin/zookeeper-server-start.sh config/zookeeper.properties
```

Sau đó bạn có thể khởi động kafka server:

```
#sudo bin/kafka-server-start.sh config/server.properties
```

3.3.7. Cài đặt Xpack

Có thể sử dụng các dòng lệnh phía dưới để cài đặt Xpack cho máy chủ:

Cài đặt Xpack cho Elasticsearch
#sudo bin/elasticsearch-plugin install x-pack

Tạo mật khẩu bằng cách chạy lệnh phía dưới (mật khẩu sẽ được tạo ra cho các user:elastic, kibana và logstash_system)

#sudo bin/x-pack/setup-passwords interactive

Cài đặt Xpack cho Kibana

#sudo bin/kibana-plugin install x-pack

Sau đó khởi động lại Elasticsearch và khởi động lại Kibana:

#sudo service elasticsearch restart

#sudo service kibana restart

3.4. Triển khai hệ thống

3.4.1. Log Sensor.

Log Sensor có một vai trò rất đơn giản trên máy chủ là theo dõi các tệp nhật ký mà những người quản trị muốn thu thập và chuyển các nội dung mới nhất của tệp nhật kí vào Kafka. Log Sensor không cần phân tích gì đối với các tập tin này để giảm thiểu tất cả những thứ phức tạp nhất có thể truyền qua kafka. Ở đây sẽ có 2 máy chủ là Web server, Server. Mỗi máy chủ sẽ được cài sẵn Logstash và có những file cấu hình như sau:

- Web server
    - Cấu hình Logstash nhận file log từ web server và chuyển dữ liệu sang máy chủ Kafka:

```
input {
   file {
     path => ["/var/log/apache2/other_vhosts_access.log"]
     type => "apache_access"
   }
}
output {
    kafka {
        bootstrap_servers => "192.168.196.132:9092"
        topic_id => "apache2"
        security_protocol => "SSL"
        ssl_truststore_location => "/home/ubuntu-02/Desktop/client.truststore.jks"
        ssl_truststore_password => "123456"
    }
   stdout { codec => rubydebug }
}
```

Hình 3. 6: File Logstash cấu hình trên web server

- Mô tả chi tiết cấu hình .
    - Input: Theo dõi sự thay đổi của log web server ở trong file có đường dẫn /var/log/apache2/other_vhosts_access.log

```
input {
  file {
    path => ["/var/log/apache2/other_vhosts_access.log"]
    type => "apache_access"
  }
}
```

    - Output: Mỗi khi có sự thay đổi thì Logstash sẽ gửi thông tin lên

```
output {
   kafka {
      bootstrap_servers => "192.168.196.132:9093"
      topic_id => "apache2"
      security_protocol => "SSL"
      ssl_truststore_location => "/home/ubuntu-
02/Desktop/client.truststore.jks"
      ssl_truststore_password => "123456"
   }
  stdout { codec => rubydebug }
}
```

Kafka với topic là apache2 và dưới sự bảo mật đường truyền bằng SSL

- Server

  - Cấu hình Logstash đọc các tệp tin log từ syslog và chuyển dữ liệu sang máy chủ Kafka:

```
input {
   file {
     path => ["/var/log/syslog"]
     type => "syslog"
   }
}
output {
    kafka {
        bootstrap_servers => "192.168.196.134:9093"
        topic_id => "syslog"
        security_protocol => "SSL"
        ssl_truststore_location => "/home/ubuntu-02/Desktop/client.truststore.jks"
        ssl_truststore_password => "123456"
    }
   stdout { codec => rubydebug }
}
```

Hình 3. 7: File Logstash cấu hình trên Syslog

  - Mô tả chi tiết cấu hình

    - Input: Theo dõi sự thay đổi của syslog ở trong tập tin có đường dẫn /var/log/syslog.

```
input {
  file {
    path => ["/var/log/syslog"]
    type => "syslog"
  }
}
```

    - Output: Mỗi khi có sự thay đổi thì Logstash sẽ gửi thông tin lên Kafka với topic là syslog và dưới sự bảo mật đường truyền bằng SSL

```
output {
   kafka {
      bootstrap_servers => "192.168.196.132:9093"
      topic_id => "syslog"
      security_protocol => "SSL"
      ssl_truststore_location => "/home/ubuntu-
02/Desktop/client.truststore.jks"
      ssl_truststore_password => "123456"
   }
   stdout { codec => rubydebug }
}
```

3.4.2. Kafka

Khởi động Zookeeper:

```
#sudo bin/zookeeper-server-start.sh config/zookeeper.properties
```

Sau đó bạn có thể khởi động kafka server:

```
#sudo bin/kafka-server-start.sh config/server.properties
```

Tạo chủ đề. Điều này có thể chạy từ bất kỳ máy nào trong hệ thống với các công cụ điều khiển kafka có sẵn. Điều quan trọng là bạn chỉ định zookeeper một cách chính xác để biết được vị trí của Kafka. Như đúng mô hình đã nêu ở phía trên chúng ta sẽ tạo ra 2 chủ đề theo từng loại log:

- Web server: Tạo ra luồng dữ liệu với topic là apache2 để kết nối web server với kafka và kafka với ELK Stack

```
#cd /opt/kafka*

#bin/kafka-topics.sh --create --zookeeper 192.168.196.132:2181 --
replication-factor 1 --partitions 1 --topic apache2
```

- Syslog: Tạo ra luồng dữ liệu với topic là syslog để kết nối server với kafka và kafka với ELK Stack

```
#cd /opt/kafka*

#bin/kafka-topics.sh --create --zookeeper 192.168.196.132:2181 --replication-factor 1 --partitions 1 –topic syslog
```

3.4.3. Log Collector – Database – Analysis tool

Log Collector sẽ đọc những thông tin thu được từ máy chủ Kafka truyền đến và chuẩn hóa, lọc thông tin dựa trên những quy tắc chuẩn hóa biết trước. Thông tin đầu ra sẽ có định dạng giống nhau và được lưu vào database, các dữ liệu ở đây đã được chuẩn hóa nên có thể sử dụng để tính toán các số liệu thống kê trên toàn hệ thống. Từ các số liệu đã được thống kê ở phía trên Analysis tool sẽ truy cập trực tiếp vào Database để hiển thị lên trình duyệt web. Trên máy chủ sẽ được cấu hình sẵn các file để tiếp nhận thông tin từ kafka và những file cấu hình đó như sau:

- Web server
  - Cấu hình Logstash để tiếp nhận thông tin log web dưới dạng JSON từ máy chủ Kafka và lưu trữ dữ liệu vào Elasticsearch:

```
input {
   kafka{
      bootstrap_servers => "192.168.196.132:9092"
      topics => "apache2"
      security_protocol => "SSL"
      ssl_truststore_location => "/home/quanghung/Desktop/client.truststore.jks"
      ssl_truststore_password => "123456"
   }
}
filter {
     grok {
        match => { "message" => ["%{IPORHOST:[remote_ip]} - %{DATA:user_name} \[%{HTTPDATE:[time]}\] \"%{WORD:[method]} %{DATA:[url]}
                                 HTTP/%{NUMBER:[http_version]}\" %{NUMBER:[response_code]}
                                %{NUMBER:[bytes]} \"%{DATA:[referrer]}\" \"%{DATA:[web_agent]}\""] }
        remove_field => "message"
      }
     mutate {
        add_field => { "read_timestamp" => "%{@timestamp}" }
      }
     date {
        match => [ "[time]", "dd/MMM/YYYY:H:m:s Z" ]
        remove_field => "[time]"
      }
}
output {
  elasticsearch { hosts => ["localhost:9200"]
      user => "elastic"
      password => "123456"
      index => "apache2-%{+YYYY.MM.dd}"
  }
  stdout { codec => rubydebug }
}
```

Hình 3. 8: File cấu hình tiếp nhận thông tin từ Web server

- Mô tả chi tiết cấu hình
  - Input: Tiếp nhận dữ liệu server Kafka với topic là apache2 và dưới sự bảo vệ đường truyền của SSL

```
input {
  kafka{
    bootstrap_servers => "192.168.196.132:9093"
    topics => "apache2"
    security_protocol => "SSL"
    ssl_truststore_location =>
"/home/quanghung/Desktop/client.truststore.jks"
    ssl_truststore_password => "123456"
  }
}
```

  - Filter: Ở đây hệ thống sẽ sử dụng plugin Grok của Logstash để phân tách các chuỗi dữ liệu nhận được. Grok hoạt động bằng cách phân tích các bản tin thành các trường riêng biệt và gán cho chúng một định danh. Hệ thống sẽ phân tách các chuỗi dữ liệu thành các trường riêng biệt để thuận tiện cho việc thống kê và tính toán khi lưu xuống Elasticsearch

```
filter {
   grok {
        match => { "message" => ["%{IPORHOST:[remote_ip]}
        - %{DATA:user_name]} \[%{HTTPDATE:[time]}\]
        \"%{WORD:[method]} %{DATA:[url]}
        HTTP/%{NUMBER:[http_version]}\"
        %{NUMBER:[response_code]}
        %{NUMBER:[bytes]} \"%{DATA:[referrer]}\"
        \"%{DATA:[web_agent]}\""] }
        remove_field => "message"
    }
   mutate {
     add_field => { "read_timestamp" => "%{@timestamp}" }
   }
  date {
     match => [ "[time]", "dd/MMM/YYYY:H:m:s Z" ]
```

Sau khi lọc và phân tích bằng Grok chúng ta sẽ thu được các trường riêng biệt lưu dưới dạng JSON như ở hình 3.9:

Hình 3. 9: Kết quả sau khi phân tích log web

- Output: Lưu dữ liệu đã được xử lý xuống Elasticsearch với tài khoản và mật khẩu.

```
output {
  elasticsearch { hosts => ["localhost:9200"]
     user => "elastic"
     password => "123456"
     index => "apache2-%{+YYYY.MM.dd}"
  }
  stdout { codec => rubydebug }
}
```

- Syslog
  - Cấu hình Logstash để tiếp nhận thông tin log web dưới dạng JSON từ máy chủ Kafka và lưu trữ dữ liệu vào Elasticsearch:

```
input {
   kafka{
     bootstrap_servers => "192.168.196.134:9093"
     topics => "syslog"
     security_protocol => "SSL"
     ssl_truststore_location => "/home/quanghung/Desktop/client.truststore.jks"
     ssl_truststore_password => "123456"
   }
}
filter{
    grok {
     match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp}
                %{SYSLOGHOST:syslog_hostname} %{DATA:syslog_program}(?:\[%{POSINT:syslog_pid}\])?:
                %{GREEDYDATA:syslog_message}" }
     add_field => [ "received_at", "%{@timestamp}" ]
     add_field => [ "received_from", "%{host}" ]
    }
    date {
     match => [ "syslog_timestamp", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]
    }
}
output {
  elasticsearch { hosts => ["localhost:9200"]
     user => "elastic"
     password => "123456"
     index => "syslog-%{+YYYY.MM.dd}"
     document_type => "syslog"

  }
  stdout { codec => rubydebug }
}
```

Hình 3. 10: File cấu hình tiếp nhận thông tin từ Syslog

- Mô tả chi tiết cấu hình
  - Input: Tiếp nhận dữ liệu server Kafka với topic là Syslog và dưới sự bảo vệ đường truyền của SSL.

```
input {
  kafka{
    bootstrap_servers => "192.168.196.132:9093"
    topics => "syslog"
    security_protocol => "SSL"
    ssl_truststore_location =>
"/home/quanghung/Desktop/client.truststore.jks"
    ssl_truststore_password => "123456"
  }
}
```

  - Filter: Ở đây hệ thống sẽ sử dụng plugin Grok của Logstash để phân tách các chuỗi dữ liệu nhận được. Grok hoạt động bằng cách phân tích các bản tin thành các trường riêng biệt và gán cho chúng một định danh. Hệ thống sẽ phân tách các chuỗi dữ liệu thành các trường riêng biệt để thuận tiện cho việc thống kê và tính toán khi lưu xuống Elasticsearch.

```
filter{
   grok {
     match => { "message" =>
"%{SYSLOGTIMESTAMP:syslog_timestamp}
                %{SYSLOGHOST:syslog_hostname}

%{DATA:syslog_program}(?:\[%{POSINT:syslog_pid}\])?:
                %{GREEDYDATA:syslog_message}" }
     add_field => [ "received_at", "%{@timestamp}" ]
     add_field => [ "received_from", "%{host}" ]
   }
   date {
     match => [ "syslog_timestamp", "MMM  d HH:mm:ss",
"MMM dd HH:mm:ss" ]
   }
}
```

Sau khi lọc và phân tích bằng Grok chúng ta sẽ thu được các trường riêng biệt lưu dưới dạng JSON như ở hình 3.11:


```
{
       "received_from" => "%{host}",
          "@timestamp" => 2017-12-23T05:11:43.000Z,
          "syslog_pid" => "31499",
     "syslog_hostname" => "louis",
    "syslog_timestamp" => "Dec 23 12:11:43",
         "received_at" => "2017-12-10T16:06:58.268Z",
            "@version" => "1",
      "syslog_program" => "postfix/smtpd",
             "message" => "2017-12-10T16:06:52.961Z jdbc Dec 23 12:11:43 louis po
stfix/smtpd[31499]: connect from unknown[95.75.93.154]",
      "syslog_message" => "connect from unknown[95.75.93.154]"
}
```


Hình 3. 11: Kết quả sau khi phân tích syslog

- Output: Lưu dữ liệu đã được xử lý xuống elasticsearch với tài khoản và mật khẩu.

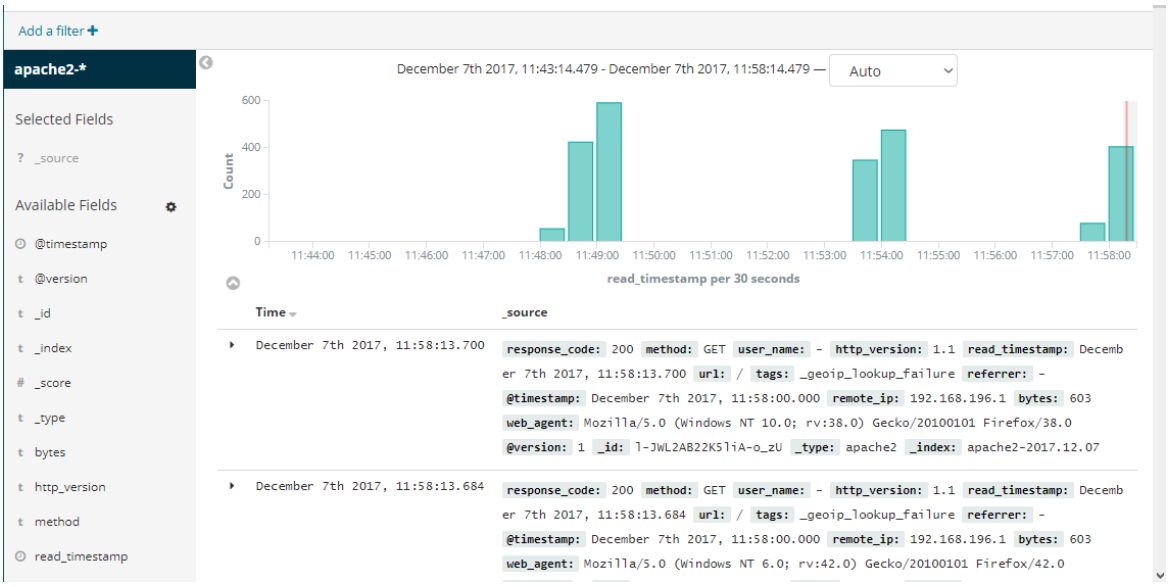
```
output {
  elasticsearch { hosts => ["localhost:9200"]
     user => "elastic"
     password => "123456"
     index => "syslog-%{+YYYY.MM.dd}"
     document_type => "syslog"

  }
  stdout { codec => rubydebug }
}
```

3.5. Kết quả thu được

Toàn bộ các quá trình của hệ thống giám sát an toàn mạng từ thu thập log, phân tích log, lưu trữ kết quả phân tích cho đến các giai đoạn chuyển tiếp log giữa các hệ thống đều đã được trình bày ở trên. Phần cuối cùng còn thiếu của một hệ thống giám sát và cũng không kém phần quan trọng đó là hiển thị các kết quả thu được sau quá trình thu thập, phân tích và lưu trữ. Dưới đây là một số các hình ảnh mô tả chi tiết về các kết quả thu được của toàn bộ quá trình giám sát hệ thống:

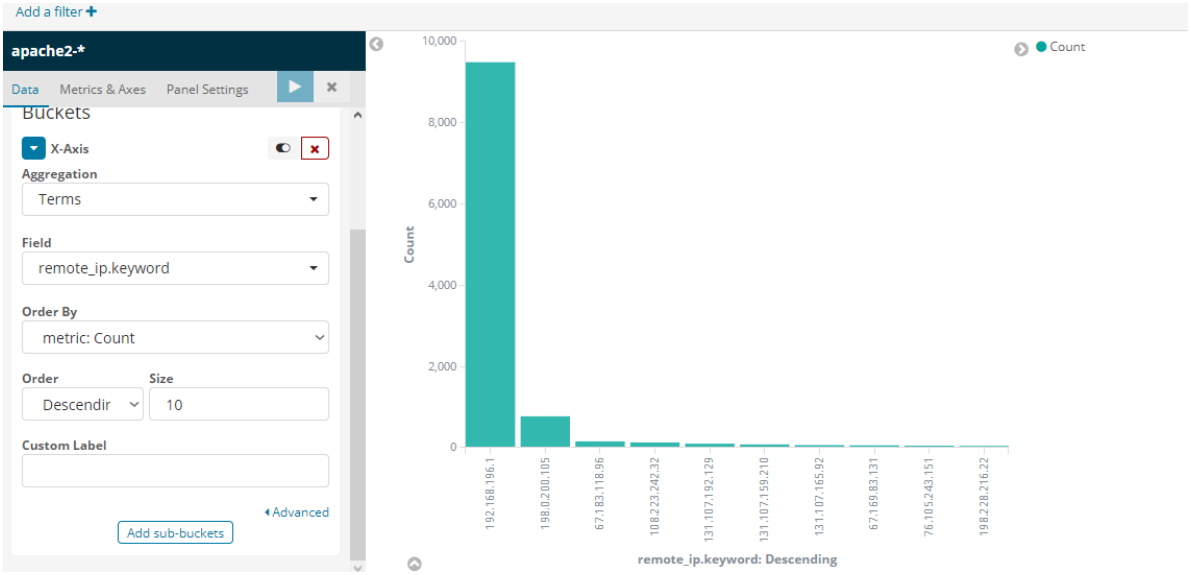
- Màn hình chính của kibana.
- Top 10 địa chỉ IP có lượt truy cập cao nhất .
- Top 5 mã lỗi được trả về nhiều nhất.
- Top 10 phiên bản trình duyệt có lượt truy câp cao nhất.
- Thống kê số lượng truy cập theo thời gian.
- Bảng điều khiển nội dung phân tích kibana.
- Bảng quản lý log trên Kibana.

- Màn hình chính của kibana hiển thị chi tiết thông tin file log kèm theo các trường đã được logstash filter từ trước. Có thể lựa chọn xem chi tiết thông tin theo các trường được hiển thị ở cột Available Fields như ở hình 3.12:



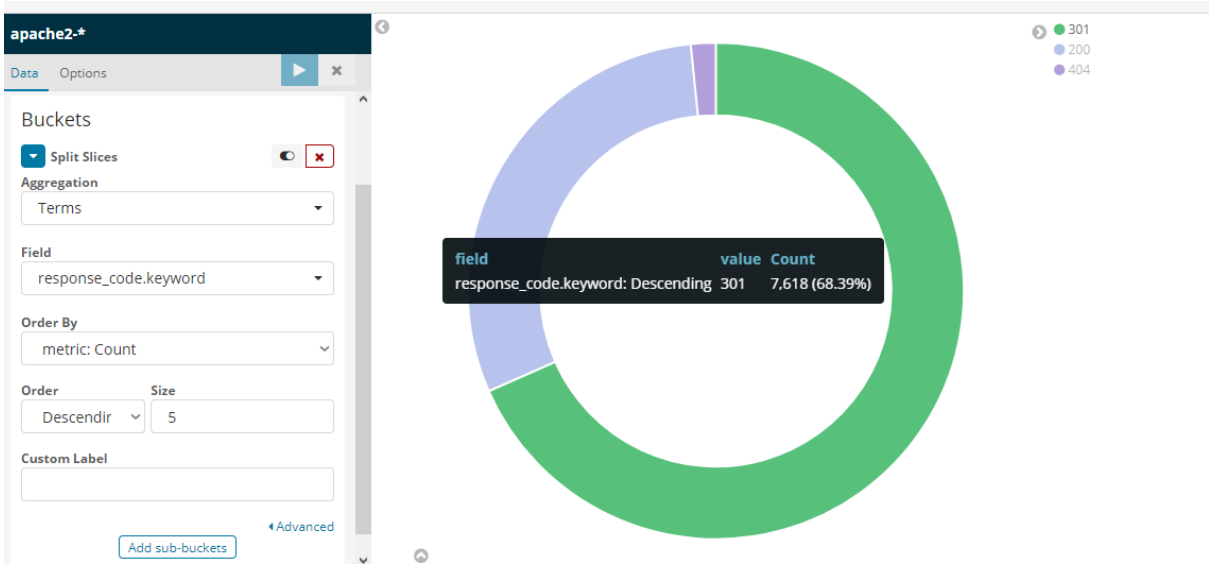
Hình 3. 12: Giao diện Kibana khi mới truy cập

- Để xem thông tin của top 10 địa chỉ IP có lượt truy cập cao nhất ở trong hệ thống có thể thực hiện như phía dưới:
  - Ở mục Aggregation ta chọn Terms
  - Ở mục Field ta chọn remote_ip.keyword
  - Ở mục Order By chọn metric: Count
  - Ở mục Order chọn Size với Descending là 10.
  - Bấm Enter và ta sẽ thu được kết quả như hình 3.13
    Nếu muốn hiển thị thêm hoặc giảm bớt đi số lượng IP thì ta có thể thay đổi ở mục Size trong Order tùy theo số lượng mình muốn.



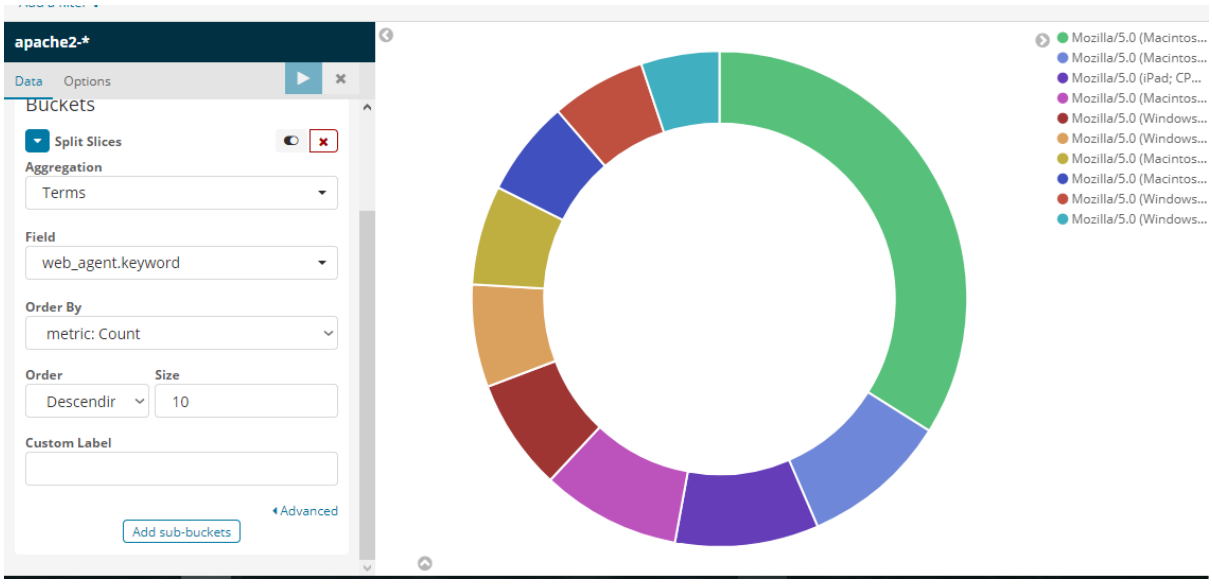
Hình 3. 13: Top 10 địa chỉ ip có lượt truy cập cao.

- Để xem thông tin của top 5 mã lỗi được trả về nhiều nhất có thể thực hiện như phía dưới:
    - Ở mục Aggregation ta chọn Terms
    - Ở mục Field ta chọn response_code.keyword
    - Ở mục Order By chọn metric: Count
    - Ở mục Order chọn Size với Descending là 5.
    - Bấm Enter và ta sẽ thu được kết quả như hình 3.14
      Nếu muốn hiển thị thêm hoặc giảm bớt đi số lượng mã lỗi thì ta có thể thay đổi ở mục Size trong Order tùy theo số lượng mình muốn.



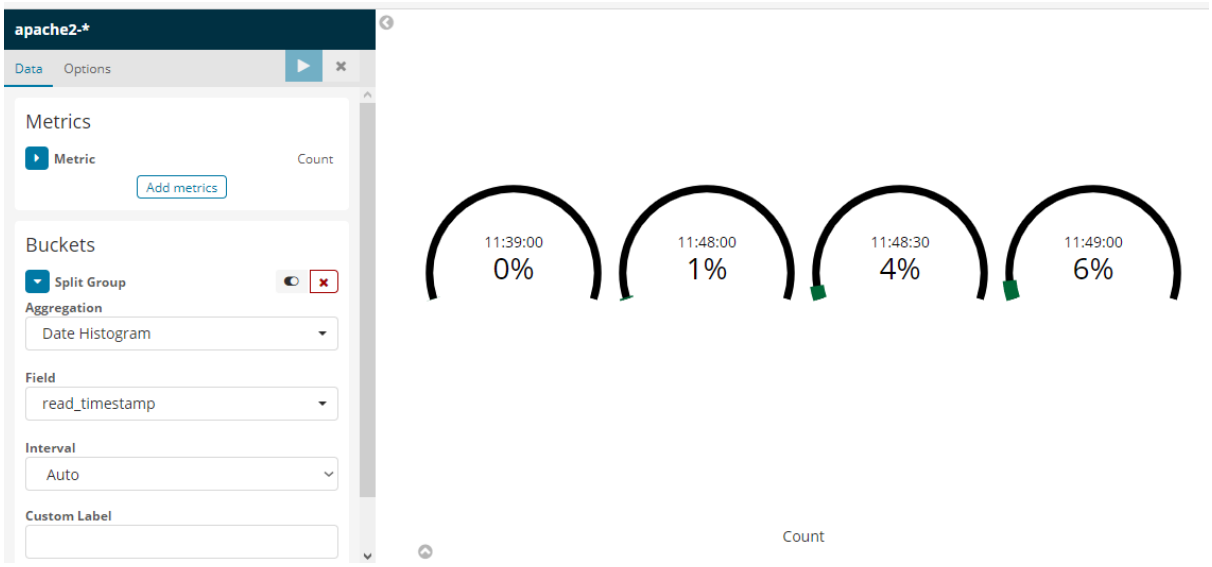
Hình 3. 14: Top 5 giá trị được phản hồi nhiều nhất.

- Để xem thông tin của top 10 phiên bản trình duyệt có lượt truy cập cao nhất có thể thực hiện như phía dưới:
    - Ở mục Aggregation ta chọn Terms
    - Ở mục Field ta chọn web_agent.keyword
    - Ở mục Order By chọn metric: Count
    - Ở mục Order chọn Size với Descending là 10.
    - Bấm Enter và ta sẽ thu được kết quả như hình 3.15
      Nếu muốn hiển thị thêm hoặc giảm bớt đi số lượng phiên bản trình duyệt thì ta có thể thay đổi ở mục Size trong Order tùy theo số lượng mình muốn.



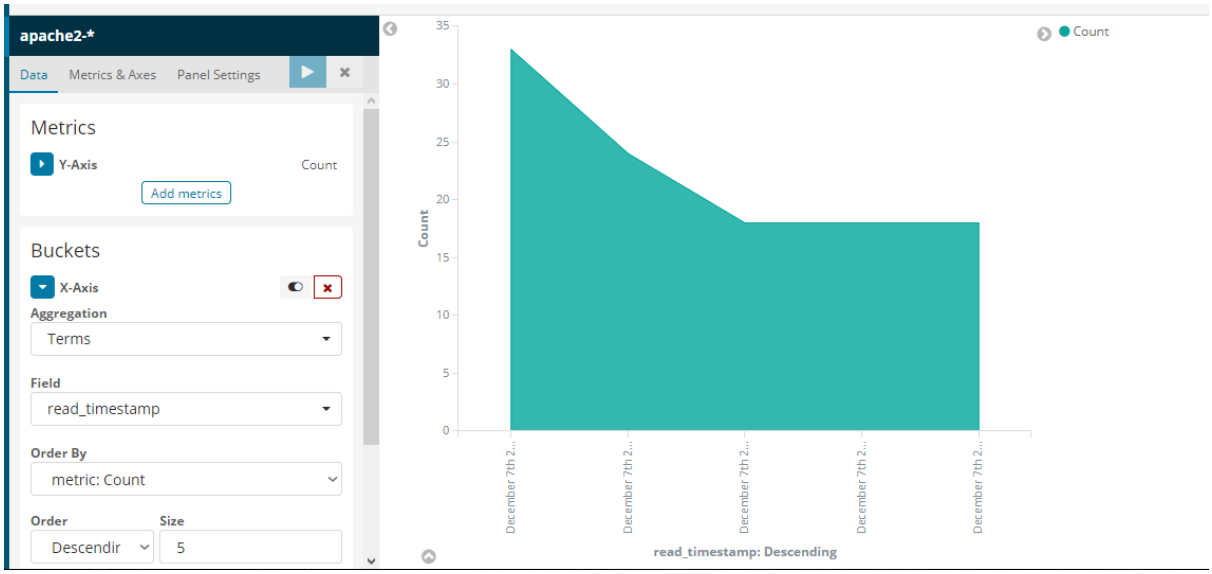
Hình 3. 15: Top 10 phiên bản trình duyệt có lượt truy cập cao.

- Để thống kê số lượng truy cập theo thời gian có thể thực hiện như phía dưới:
  - Ở mục Aggregation ta chọn Date Histogram
  - Ở mục Field ta chọn read_timestamp
  - Ở mục Interval chọn Auto
  - Bấm Enter và ta sẽ thu được kết quả như hình 3.16



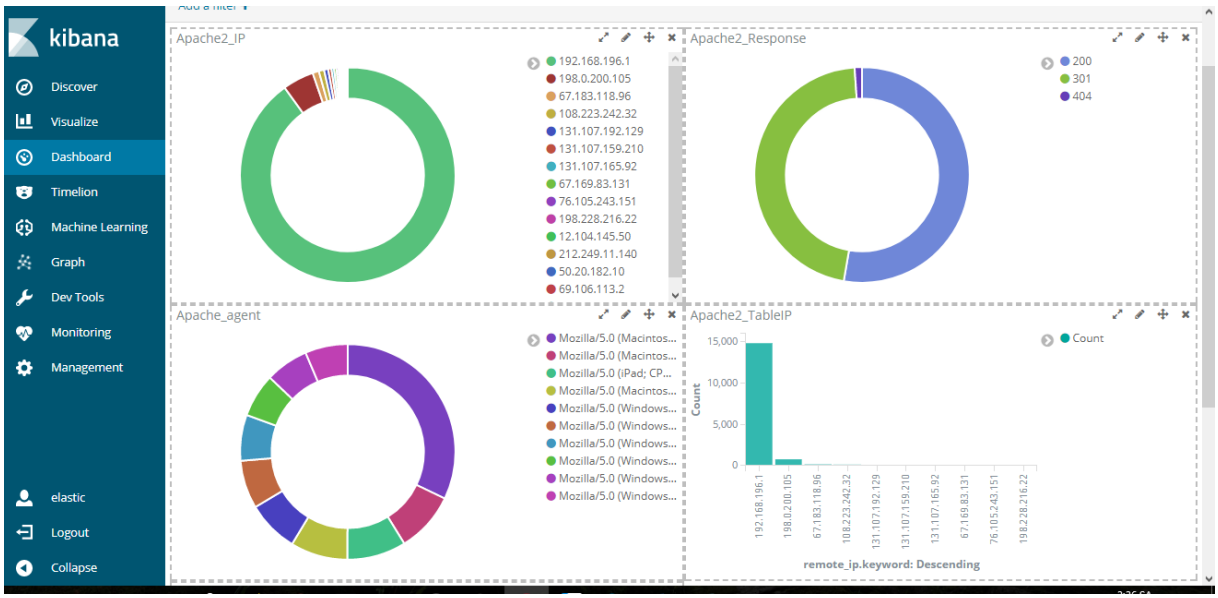
Hình 3. 16: Thống kê số lượng truy cập theo thời gian

- Để thống kê số lượng truy cập theo thời gian có thể thực hiện như phía dưới:
  - Ở mục Aggregation ta chọn Term
  - Ở mục Field ta chọn read_timestamp
  - Ở mục Order By chọn metric: Count
  - Ở mục Order chọn Size với Descending là 5.
  - Bấm Enter và ta sẽ thu được kết quả như hình 3.17



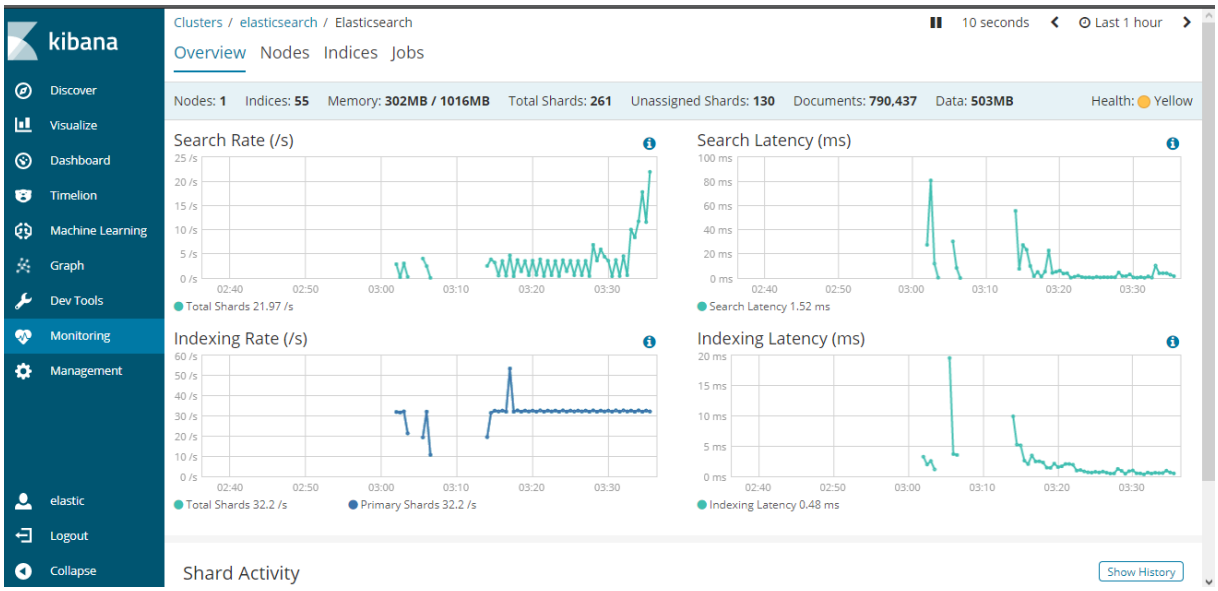
Hình 3. 17: Thống kê số lượng truy cập theo thời gian

- Để thuận tiện cho việc theo dõi thông số của toàn bộ hệ thống ta vào Dashboard ở bên thanh bên tay trái của màn hình. Kết quả sẽ được hiển thị như hình 3.18:



Hình 3. 18: Bảng điều khiển nội dung phân tích kibana

- Để thuận tiện cho việc theo dõi thông số của các Node trong hệ thống ta vào Monitoring ở bên thanh bên tay trái của màn hình. Kết quả sẽ được hiển thị như hình 3.19:



Hình 3. 19: Bảng quản lý log trên kibana

3.6. Kết chương

Trong chương 3 đã áp dụng được các giải pháp đề xuất ở chương 2 để xây dựng một hệ thống giám sát an toàn mạng nhằm theo dõi, giám sát. Cấu hình và triển khai thành công hệ thống trên máy ảo. Thử nghiệm hệ thống và đưa ra được một số kết quả phục vụ cho việc phân tích và giám sát hệ thống

.

KẾT LUẬN

Kết quả đạt được:

Từ nội dung của 3 chương, đồ án đạt được những kết quả quan trọng sau:

- Trình bày được kiến trúc tổng quan của một hệ thống giám sát an toàn mạng. Trong đó có nói chi tiết về cách thức hoạt động của từng thành phần trong một hệ thống giám sát an toàn mạng, cũng như đưa ra được các công cụ thường được sử dụng trong hệ thống giám sát an toàn mạng
- Đồ án cũng đưa ra được các giải pháp để xây dựng một hệ thống giám sát an toàn mạng. Đặc biệt tập trung nghiên cứu để đưa ra các biện pháp khắc phục được các nhược điểm của mô hình. Trong quá trình nghiên cứu đồ án cũng giới thiệu được mô hình tổng quan của hệ thống cũng như cách thức hoạt động của từng thành phần
- Xây dựng thành công một hệ thống giám sát an toàn mạng sử dụng trong việc theo dõi, phát hiện và đưa ra các mô hình, biểu đồ ở trong hệ thống. Từ kết quả thử nghiệm thu được, đồ án đã rút ra được ưu điểm, nhược điểm của mô hình và có được những hướng phát triển, cải thiện hệ thống giám sát an toàn mạng trong tương lai.

Hướng phát triển tiếp trong tương lai:

Từ các kiến thức đã tìm hiểu và các kinh nghiệm thu được trong quá trình xây dựng hệ thống giám sát an toàn mạng, đồ án nhận thấy những hướng nghiên cứu và phát triển trong tương lai:

- Tiếp tục thử nghiệm hệ thống giám sát an toàn mạng với nhiều các hệ thống cùng kết nối đến hệ thống xử lý. Từ đó, ta có thể tinh chỉnh, cài đặt lại các ứng dụng trong hệ thống để sao cho đạt hiệu suất làm việc cao nhất. Hệ thống cần được tối ưu để có thể tận dụng tối đa hiệu năng phân cứng của máy tính và mạng. Điều này có ý nghĩa cực kỳ quan trọng để có thể đảm bảo được quá trình xử lý dữ liệu theo thời gian thực của hệ thống
- Nghiên cứu kết hợp mô hình với các ứng dụng khác để nhằm nâng cao hiệu suất làm việc cũng như tính bảo mật của hệ thống
- Nghiên cứu mở rộng hệ thống với nhiều các máy chủ, các đầu vào khác nhau để có thể tinh chỉnh, áp dụng hệ thống cho các doanh nghiệp lớn

TÀI LIỆU THAM KHẢO

[1] Bài giảng kỹ thuật theo dõi giám sát an toàn mạng, Nguyễn Ngọc Điệp, 2015

[2] Introduction into the ELK stack, Alexander Reelsen, November 2014

[3] Forays into Kafka -Logstash transport, https://www.rittmanmead.com/blog/. [Đã truy cập 11/2017].

[4] Introduction - Apache Kafka - The Apache Software Foundation, https://kafka.apache.org/intro. [Đã truy cập 11/2017]

[5] The Best of Apache Kafka Architecture, Ranganathan Balashanmugam

[6] Kafka: a Distributed Messaging System for Log Processing, Jay Kreps - Neha Narkhede - Jun Rao

[7] Just Enough Kafka for the Elastic Stack, https://www.elastic.co/blog/. [Đã truy cập 12/2017]

[8] Intro to ELK and Integrate Kafka with ELK, https://cloudbot.blogspot.com/2017/. [Đã truy cập 11/2017]