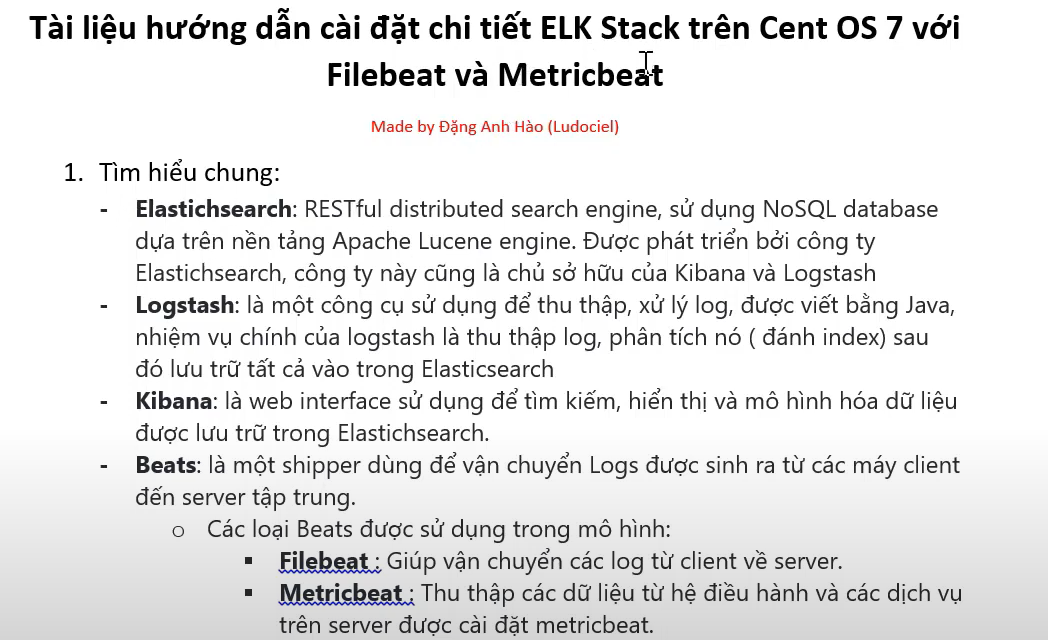
# ELK

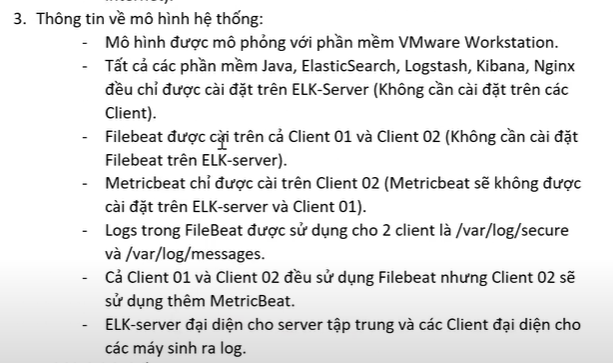


c. phần mềm

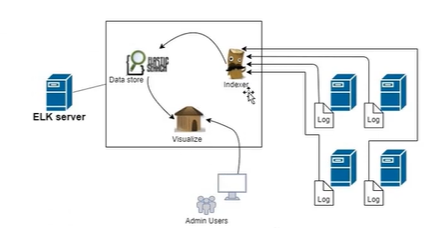
Java 8 được yêu cầu bời Logstash và Elasticsearch, các phiên bản java khác ko có tác dụng

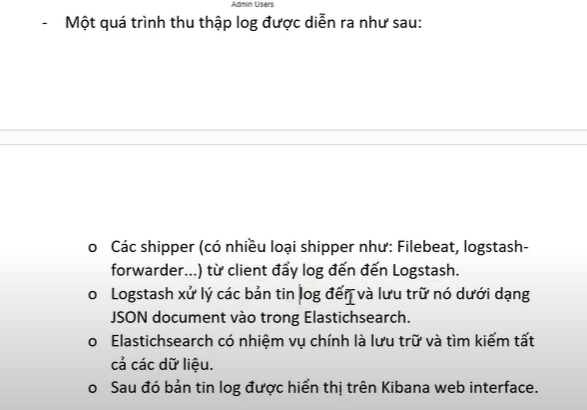
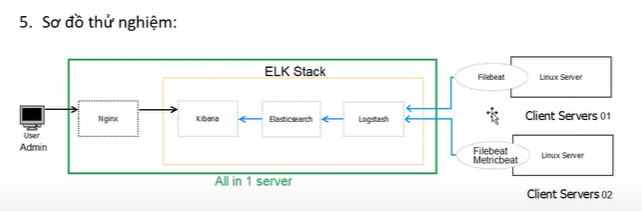
Logstash, Elasticsearch, Kibana, Filebeat và Metricbeat đều dùng chung version

Nginx để làm reserve proxy cho Kibana (Nginx ủy quyền cho Kibana để Kibana có thể được truy cập qua các Web browser ngoài internet)



* Mô hình

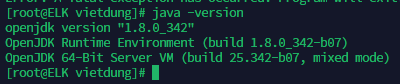


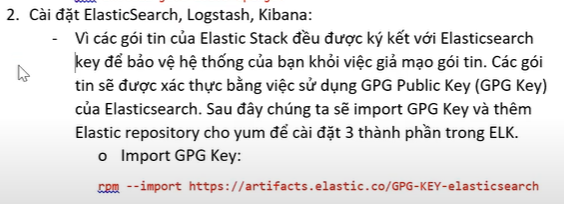
 

## Cài đặt

* Cài java 8 JDK

yum -y install java-openjdk-devel java-openjdk



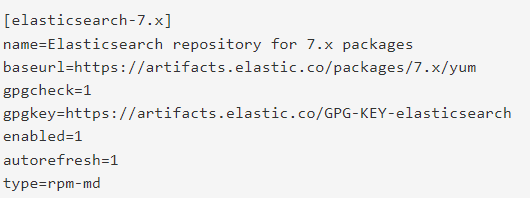


rpm --import <https://artifacts.elastic.co/GPG-KEY-elasticsearch>

* Thêm repository for elastichSearch

vim /etc/yum.repos.d/elasticsearch.repo

thêm đoạn này vào



Cách để thêm nhanh:

cat >> /etc/yum.repos.d/elasticsearch.repo << EOF

[elasticsearch-7.x]

name=Elasticsearch repository for 7.x packages

baseurl=https://artifacts.elastic.co/packages/7.x/yum

gpgcheck=1

gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch

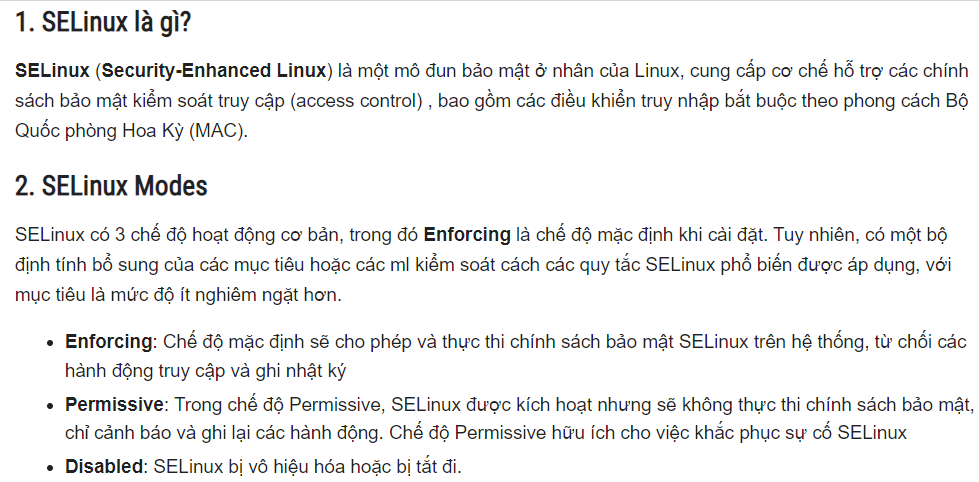
enabled=1

autorefresh=1

type=rpm-md

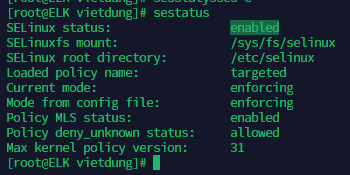
EOF

* Tắt Selinux



* Check Selinux

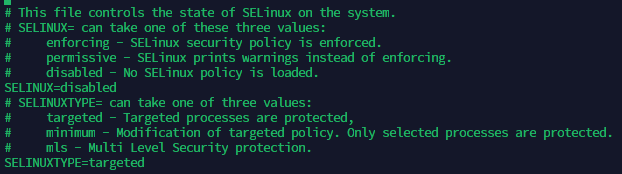
sestatus



* Tắt se

sed -i 's/\(^SELINUX=\).\*/\SELINUX=disabled/' /etc/sysconfig/selinux

hooặc vào **/etc/sysconfig/selinux** đổi thành disable



Reboot lại máy và check lại sestatus

## Cài đặt và cấu hình ElastichSearch

yum clean all && yum makecache && yum -y install elasticsearch

* Sửa đổi file config của elastichSearch

nano /etc/elasticsearch/elasticsearch.yml



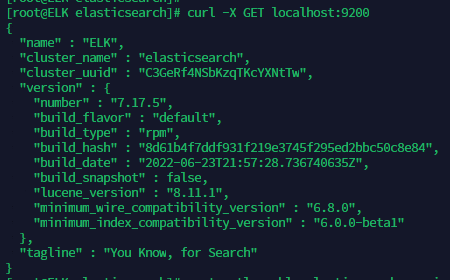
#network.host: 192.168.0.1 -> network.host: localhost

* Start elasticsearch

systemctl restart elasticsearch.service

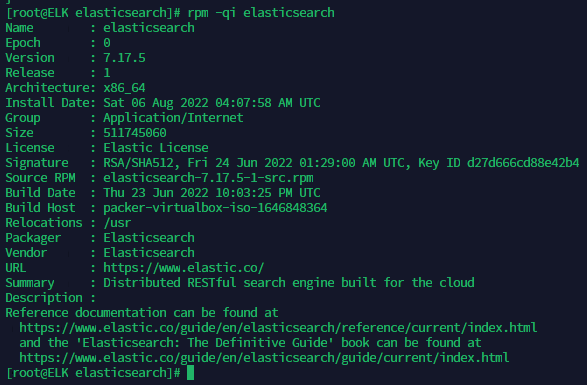
systemctl enable elasticsearch.service

enable tức là để khi start máy thì elasticsearch cũng sẽ được start cùng luôn, giống trong Startup của Task Manager window



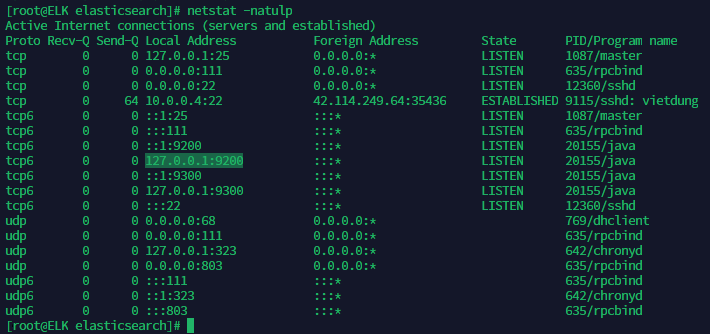
Oke, đã install xog

rpm -qi elasticsearch



Nếu có bất kỳ thay đổi nào liên quan đến ELK, thì phải **restart** Elasticsearch

netstat -natulp

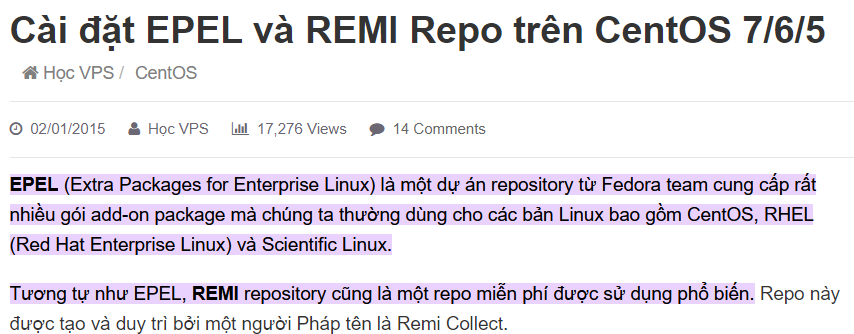


## 2. cài đặt và cấu hình Kibana

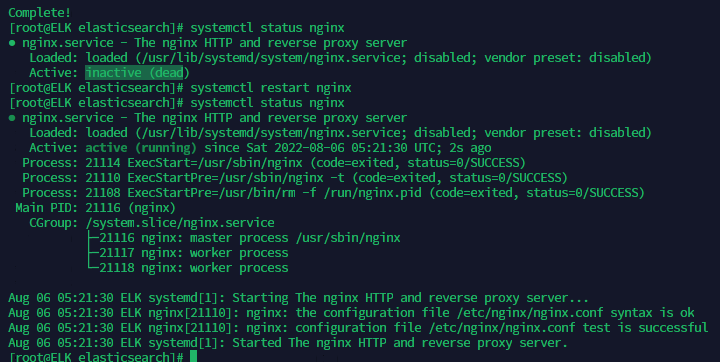
Install Proxy nginx trước

yum install epel-release

yum install nginx



Sau khi cài xong nginx



systemctl restart nginx

systemctl enable nginx (để nginx bật mỗi khi khởi động máy)\_

Check firewall

sudo firewall-cmd --state



* Cài đặt Kibana

yum -y install kibana

config kibana

nano /etc/kibana/kibana.yml

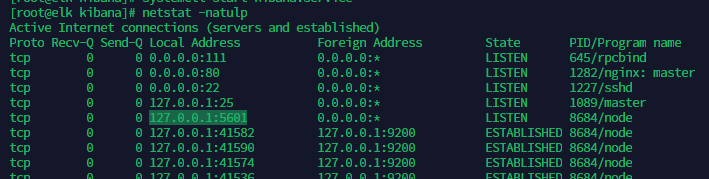
#server.host: "localhost" -> server.host: "localhost"



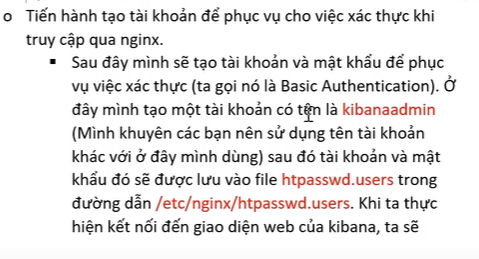
systemctl enable kibana.service

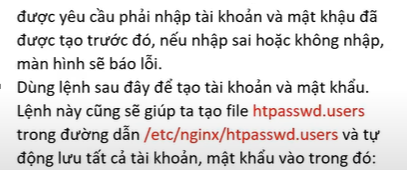
systemctl start kibana.service

để vào được kibana qua web browser cần có nginx



Kibana hđ trên port 5601





* Tạo tk, mk để truy nhập ELK server (thiết lập cho nginx để client access ELK cần phải TK, MK)

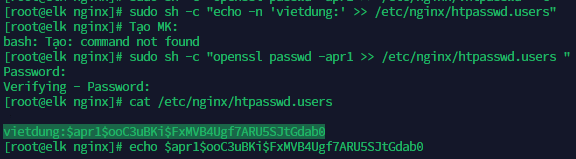
Tạo TK:

sudo sh -c "echo -n 'vietdung:' >> /etc/nginx/htpasswd.users"

Tạo MK:

sudo sh -c "openssl passwd -apr1 >> /etc/nginx/htpasswd.users "

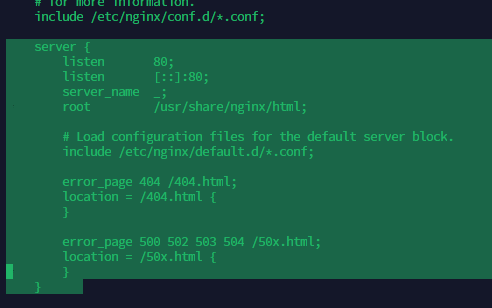
xác thực qua file **htpasswd.users** hoặc **.htpasswd**

****

Oke

* Xóa server block trong nginx

nano /etc/nginx/nginx.conf



Xóa hết đoạn bôi đen

* Tạo file config mới

nano /etc/nginx/conf.d/kibana.conf

server {

    listen 80;

    server\_name elk.co;

    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;

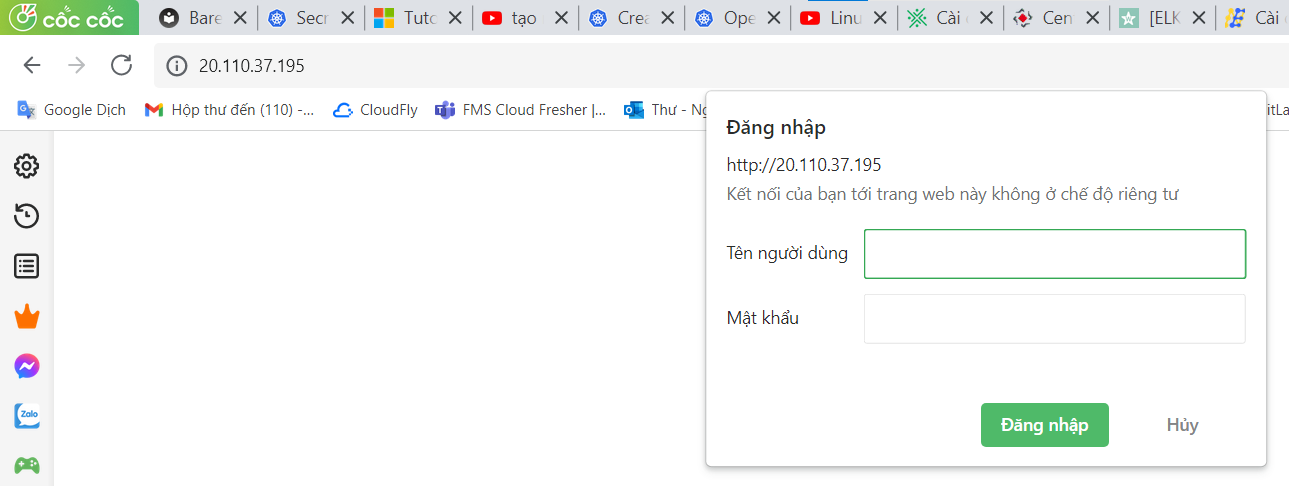
    }

}

File này sẽ giúp điều hướng vào kibana-ELK cần phải xác thực

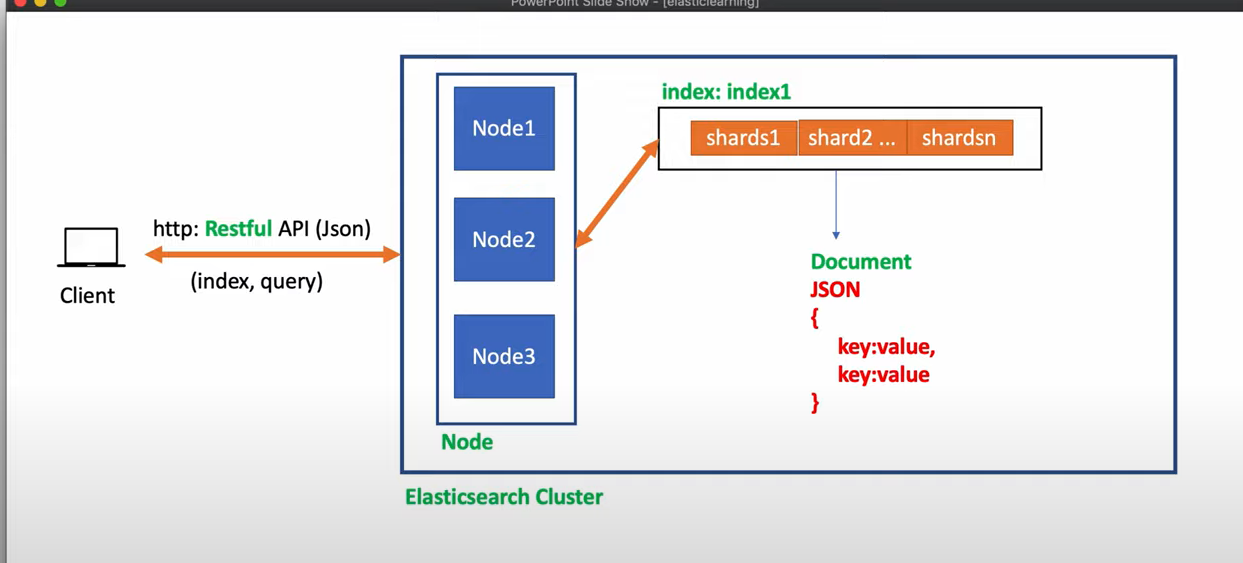
systemctl restart nginx

bgio access, truy nhập



User/pass vừa tạo ở htpasswd/users

# Elasticsearch tiep



Ta có 1 hoặc 1 cụm ELK trên node 1 2 3

Là 1 hệ thống phân tán: khi tạo 1 elasticSearch từ 1 node ban đầu thì n đồng thời tạo ra 1 ELK server và 1 ELK cluster,

Sau đó ta có thể cho phép các ELK khác kết nối vào hệ thống của ta

 dù bạn chỉ tạo ra hệ thống với 1 server (node) thì vẫn có một cluster, sau này có thể nối nhiều server phân tán vào cluster để mở rộng khả năng của hệ thống mà ở đó một node (server) có thể có một chức năng riêng như (master node, data, client...).

Index là 1 tập hợp dữ liệu, tương tự như table, bảng dữ liệu trong SQL

Tài liệu lưu trong index là các document. Định dạng dưới dạng json, docs là định dạng dữ liệu nhỏ nhất trong ELK server

Docker-compose.yaml

version: '3'

services:

  elasticsearch:

    image: docker.elastic.co/elasticsearch/elasticsearch:7.0.1

    container\_name: elasticsearch

    environment:

      - node.name=elasticsearch

      - cluster.name=datasearch

      - bootstrap.memory\_lock=true

      - "ES\_JAVA\_OPTS=-Xms512m -Xmx512m"

      - cluster.initial\_master\_nodes=elasticsearch

    ulimits:

      memlock:

        soft: -1

        hard: -1

    ports:

      - "9200:9200"

    volumes:

      - esdata:/home/thuytt/elk/data

  kibana:

    image: docker.elastic.co/kibana/kibana:7.0.1

    ports:

      - "5601:5601"

volumes:

  esdata:

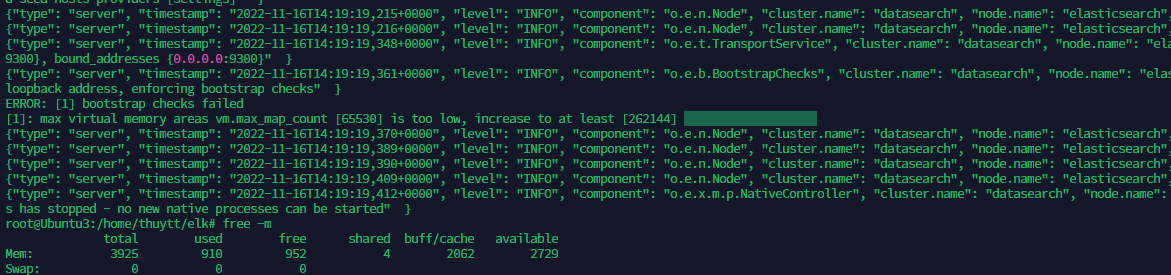
    driver\_opts:

      device: /home/thuytt/elk/data1

      o: bind

      type: tmpfs

lưu ý phải tạo volume trước



Lỗi ko start được elastic vì cái count quá thấp

Fix:

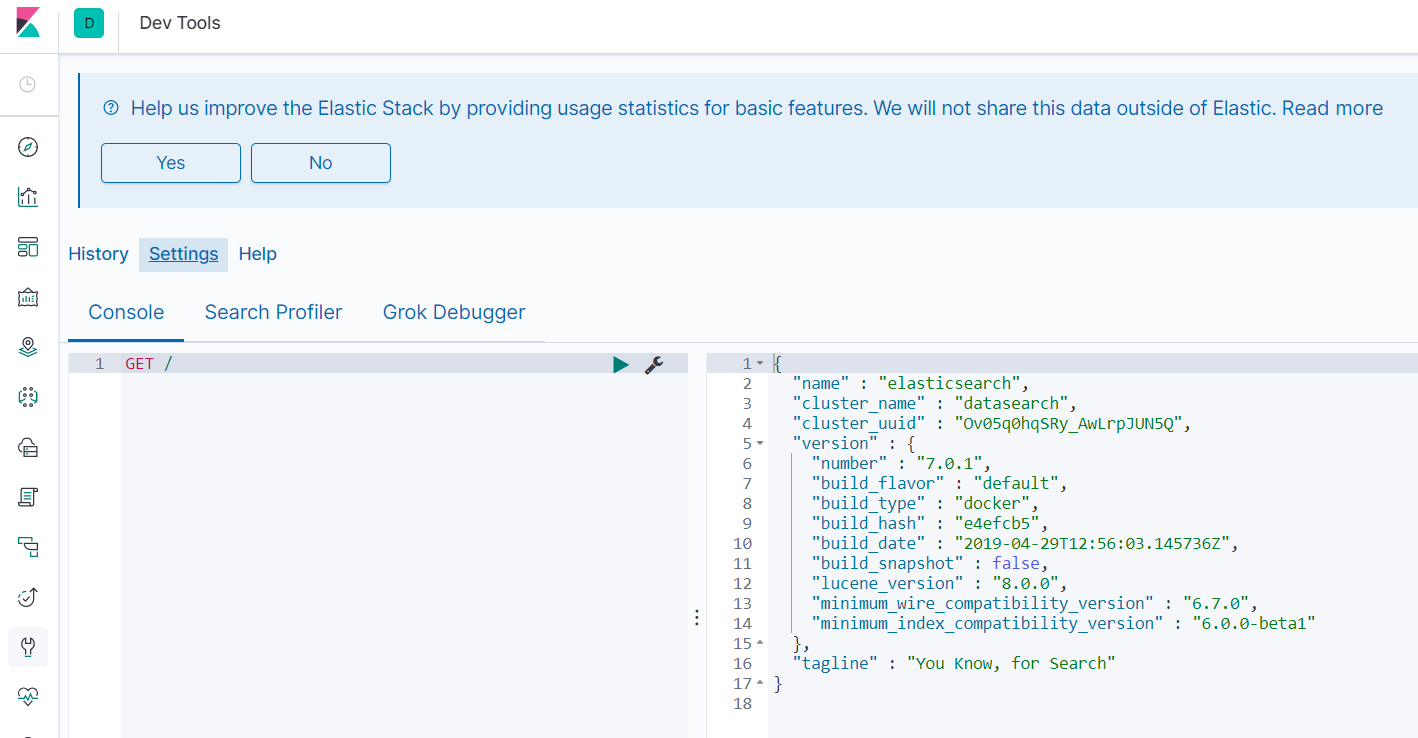
sysctl -w vm.max\_map\_count=262144

check lại:

curl -X GET localhost:9200

vào kibana port 5601

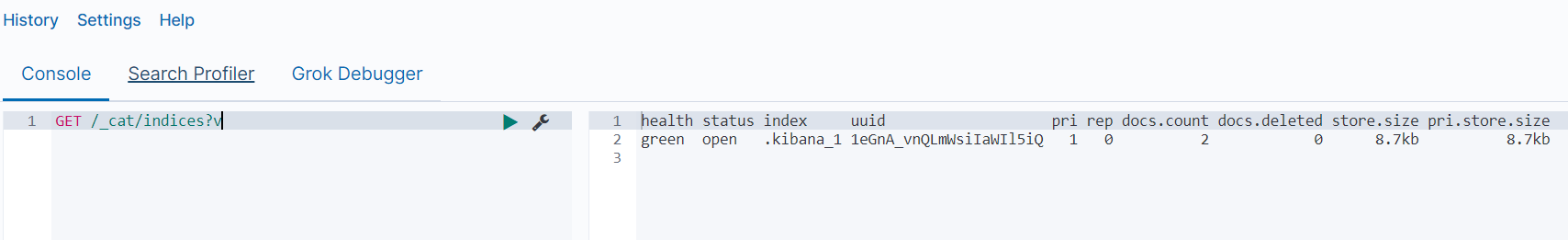
* Dev tool (góc bên trái màn hình)

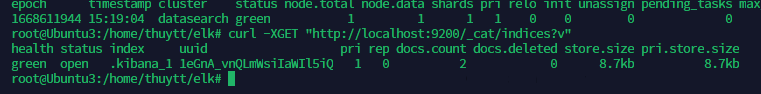


Oke cài thành công ELK và kibana

ELK có 3 trạng thái : xanh, vàng, đỏ

* Xanh, vàng là bthuong
* Check index





Chỉ có 1 index ở trạng thái green với 2 docs

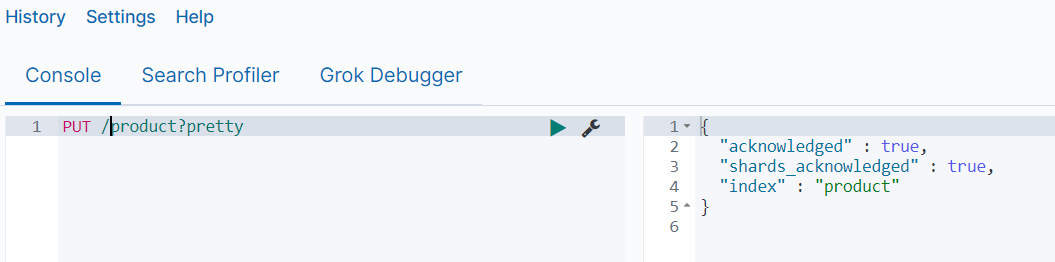
Thêm ?v sao mỗi trường truy vấn

* Tạo thêm index : 1 index có tên là product

Dùng kibana

PUT /product?pretty

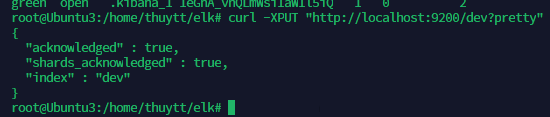
Index tên product, thêm ?pretty để hiển thị thông tin index

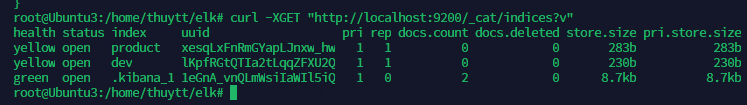


Oke giờ đã có 2 index

Ta cũng có thể thêm index trên terminal bằng lệnh

curl -XPUT "http://localhost:9200/dev?pretty"

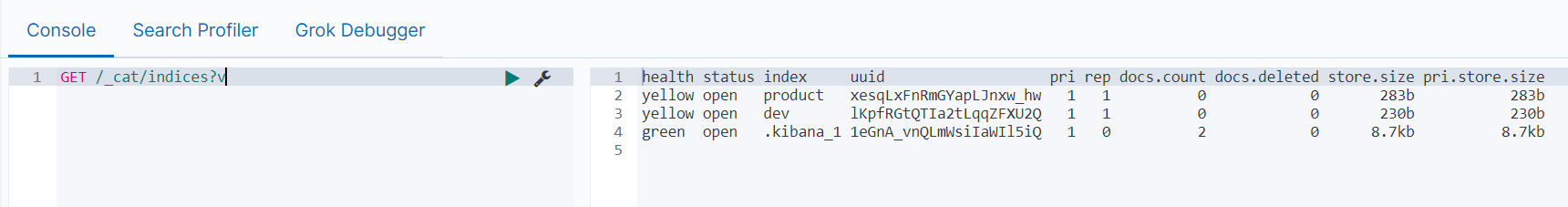




Ta cũng có thể kiểm tra trên kibana bằng lệnh

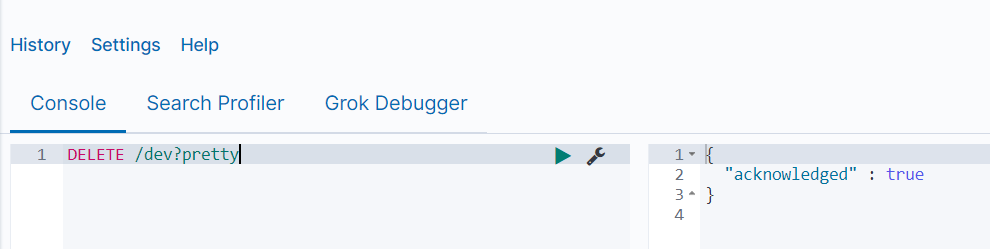
GET /\_cat/indices?pretty

GET /\_cat/indices?v



* Xóa index : index dev

DELETE /dev?pretty



* Tạo dữ liệu vào 1 index

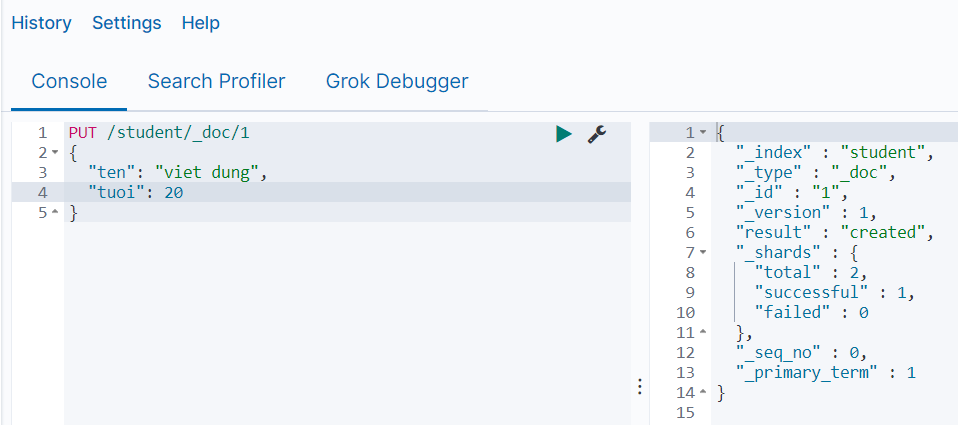
PUT /student/\_doc/1      # tạo 1 cơ sở dữ liệu có id=1 vào doc của index student

{

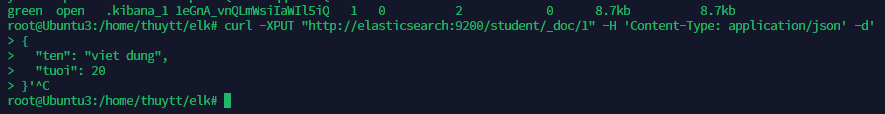
  "ten": "viet dung",

  "tuoi": 20

}

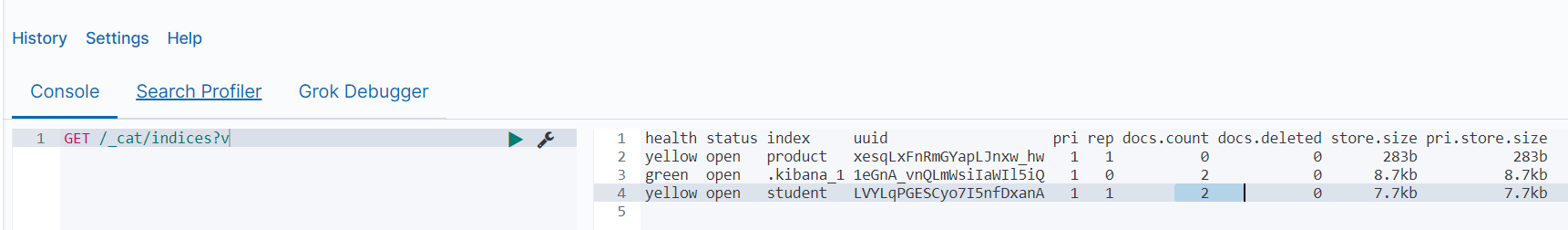


Copy Url này từ kibana ra termius ta có



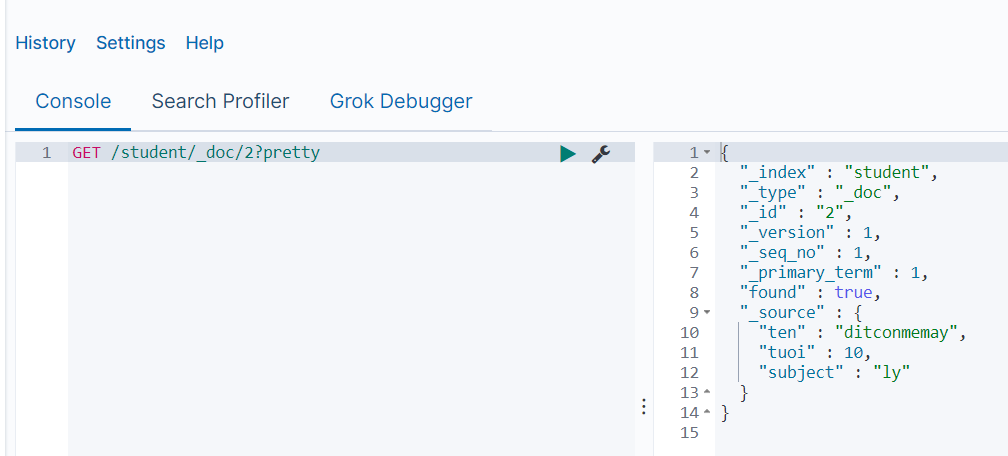
Khi tạo thêm 1 id nữa là id 2 thì check đã có 2 docs.count trong index student

Docs.count = 2, tức trong



* Check dữ liệu trong 1 docs.count

GET /student/\_doc/2?pretty



Để update dữ liệu trong docs, ta dùng lệnh PUT, PUT đè vào dữ liệu cũ được

## Tạo mới, cập nhật nhiều docs cho các index

POST /\_bulk    # sau đó là các dữ liệu của docs, mỗi docs được mô tả bởi 2 dòng

{"index": {"\_index": "student", "\_id": 4}}       # thông tin index của dữ liệu

{"ten": "dung", "age":20}                        # nội dung của docs (2 cặp 1)

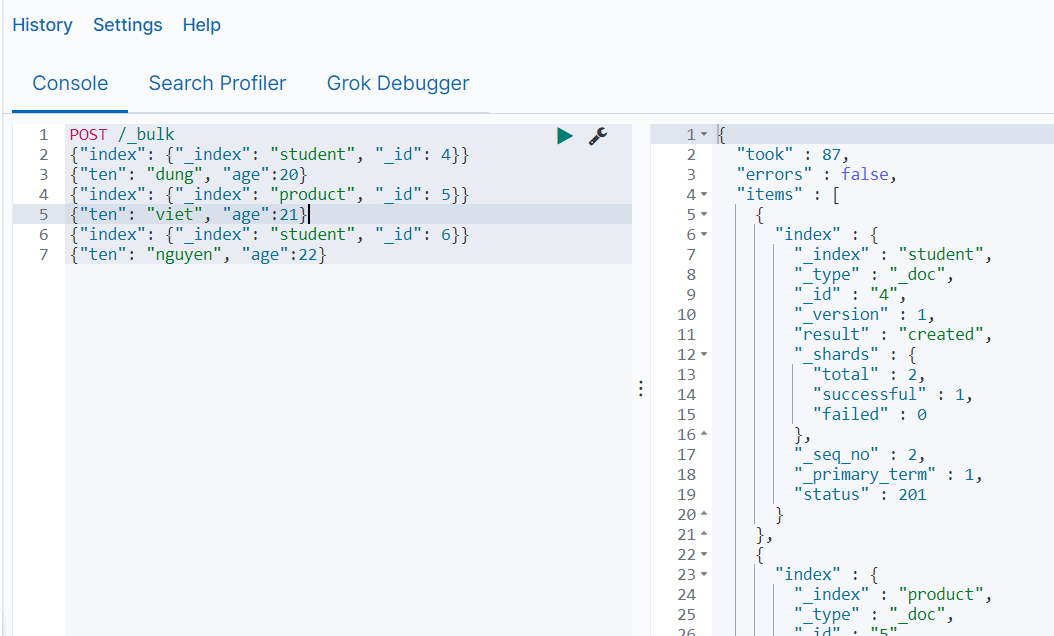
{"index": {"\_index": "product", "\_id": 5}}

{"ten": "viet", "age":21}

{"index": {"\_index": "student", "\_id": 6}}

{"ten": "nguyen", "age":22}

Câu lệnh này có thể tạo hoặc update



Ta còn 1 cách nữa là lưu nội dung này ra cách file để làm cho chuyên nghiệp

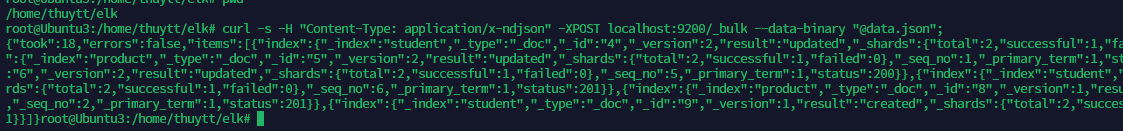
Tạo 1 file data.json



curl -s -H "Content-Type: application/x-ndjson" -XPOST localhost:9200/\_bulk --data-binary "@data.json";

* Hoặc

curl -s -H "Content-Type: application/x-ndjson" -XPOST localhost:9200/\_bulk --data-binary "@/home/thuytt/elk/data.json";



Tạo 1 ansible cho ELK

* Inventory
* Playbook

## Install logstash trên con server khác để thu thập log

Tôi cài file-beat trên server này để thu thập log trên server này và gửi về server ELK

* **Packetbeat** : lấy / gửi các gói tin mạng
* **Filebeat** : lấy / gửi các file log của Server
* **Metricbeat** : lấy / gửi các log dịch vụ (Apache log, mysql log ...)

Server ELK

* Cài elastic
* Cài kibana
* Cài logstash

Logstash có 3 thành phần cấu hình

* Cấu hình input: file cấu hình tại /etc/logstash/conf.d/02-beats-input.conf, phần này sẽ cấu hình để nó nhân đầu vào do Beats gửi đến cổng beats, thực hiện lệnh sau để tạo file 02-beats-input.conf

echo 'input {

beats {

host => "0.0.0.0"

port => 5044

}

}' > /etc/logstash/conf.d/02-beats-input.conf

* Cấu hình đầu ra, file cấu hình tại /etc/logstash/conf.d/30-elasticsearch-output.conf, phần này sẽ cấu hình sau khi Logstash nhận dữ liệu đầu vào từ Beats, nó xử lý rồi gửi đến Elasticsearch (localhost:9200). Thực hiện lệnh để tạo file 30-elasticsearch-output.conf

Định dạng lại dữ liệu

echo 'output {

elasticsearch {

hosts => ["localhost:9200"]

manage\_template => false

index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"

}

}' > /etc/logstash/conf.d/30-elasticsearch-output.conf

* Thứ 3 là định dạng lại dữ liệu dễ đọc hơn, ko cần thiết lập cũng dk

Ngoài ra nếu muốn lọc các log, định dạng lại các dòng log ở dạng dễ đọc, dễ hiểu hơn thì cấu hình filter tại file /etc/logstash/conf.d/10-syslog-filter.conf, ví dụ sau là cấu hình định dạng lại cấu trúc system log, lấy theo hướng dẫn tại document của Logstash

echo 'filter {

  if [fileset][module] == "system" {

    if [fileset][name] == "auth" {

      grok {

        match => { "message" => ["%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?: %{DATA:[system][auth][ssh][event]} %{DATA:[system][auth][ssh][method]} for (invalid user )?%{DATA:[system][auth][user]} from %{IPORHOST:[system][auth][ssh][ip]} port %{NUMBER:[system][auth][ssh][port]} ssh2(: %{GREEDYDATA:[system][auth][ssh][signature]})?",

                  "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?: %{DATA:[system][auth][ssh][event]} user %{DATA:[system][auth][user]} from %{IPORHOST:[system][auth][ssh][ip]}",

                  "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?: Did not receive identification string from %{IPORHOST:[system][auth][ssh][dropped\_ip]}",

                  "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sudo(?:\[%{POSINT:[system][auth][pid]}\])?: \s\*%{DATA:[system][auth][user]} :( %{DATA:[system][auth][sudo][error]} ;)? TTY=%{DATA:[system][auth][sudo][tty]} ; PWD=%{DATA:[system][auth][sudo][pwd]} ; USER=%{DATA:[system][auth][sudo][user]} ; COMMAND=%{GREEDYDATA:[system][auth][sudo][command]}",

                  "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} groupadd(?:\[%{POSINT:[system][auth][pid]}\])?: new group: name=%{DATA:system.auth.groupadd.name}, GID=%{NUMBER:system.auth.groupadd.gid}",

                  "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} useradd(?:\[%{POSINT:[system][auth][pid]}\])?: new user: name=%{DATA:[system][auth][user][add][name]}, UID=%{NUMBER:[system][auth][user][add][uid]}, GID=%{NUMBER:[system][auth][user][add][gid]}, home=%{DATA:[system][auth][user][add][home]}, shell=%{DATA:[system][auth][user][add][shell]}$",

                  "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} %{DATA:[system][auth][program]}(?:\[%{POSINT:[system][auth][pid]}\])?: %{GREEDYMULTILINE:[system][auth][message]}"] }

        pattern\_definitions => {

          "GREEDYMULTILINE"=> "(.|\n)\*"

        }

        remove\_field => "message"

      }

      date {

        match => [ "[system][auth][timestamp]", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]

      }

      geoip {

        source => "[system][auth][ssh][ip]"

        target => "[system][auth][ssh][geoip]"

      }

    }

    else if [fileset][name] == "syslog" {

      grok {

        match => { "message" => ["%{SYSLOGTIMESTAMP:[system][syslog][timestamp]} %{SYSLOGHOST:[system][syslog][hostname]} %{DATA:[system][syslog][program]}(?:\[%{POSINT:[system][syslog][pid]}\])?: %{GREEDYMULTILINE:[system][syslog][message]}"] }

        pattern\_definitions => { "GREEDYMULTILINE" => "(.|\n)\*" }

        remove\_field => "message"

      }

      date {

        match => [ "[system][syslog][timestamp]", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]

      }

    }

  }

}

' > /etc/logstash/conf.d/10-syslog-filter.conf

Bây giờ t chỉ cần cài file-beat trên con khác để lắng nghe log trên chính con đó

Lắng nghe mọi dữ liệu (beat) từ các con server khác

Trên con agent

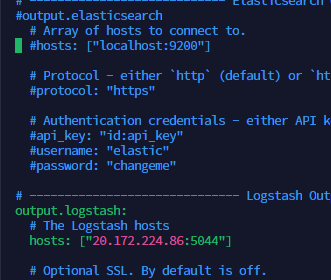
Cài đặt: file-beat

sudo apt install filebeat

nano /etc/filebeat/filebeat.yml

command #output.elasticsearch

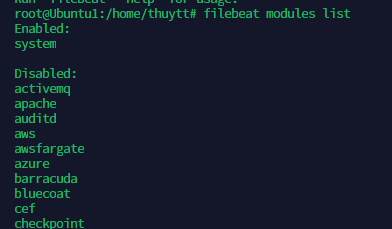
tức ko cho n gửi trực tiếp đến elasticsearch, phải cho n gửi đến logstash trước



IP kia là IP của con logstash server, chính là con ELK server

Check những thành phần đang được filebeat lắng nghe log

filebeat modules list



Hiện tại mới enable lắng nghe logs của system

* Command kích hoạt thu thập logs

filebeat modules enable system

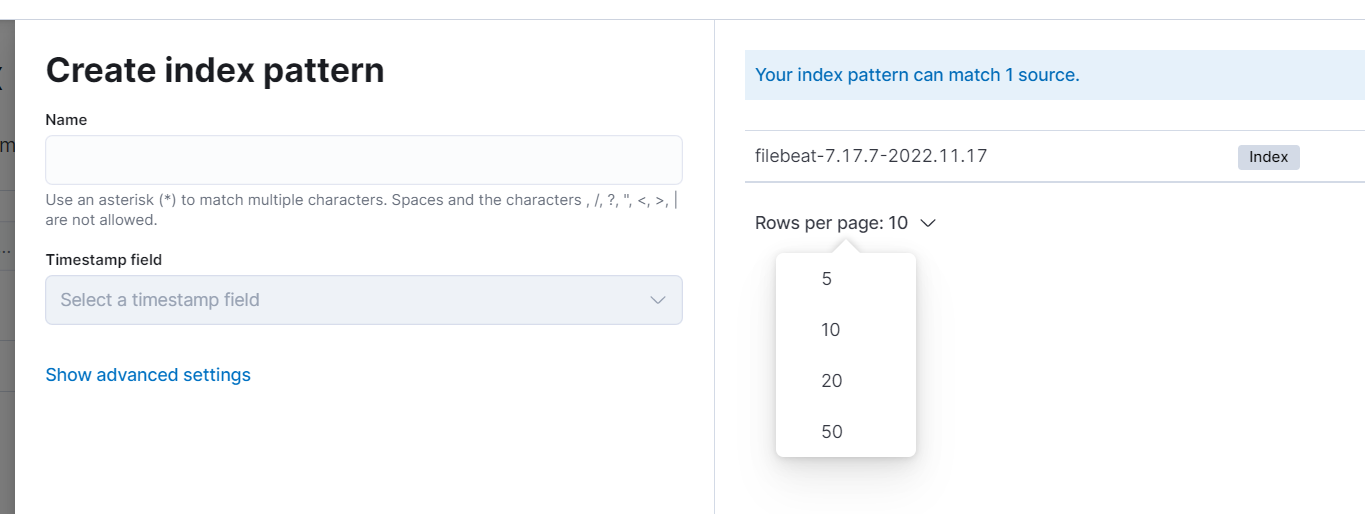
filebeat modules enable apache

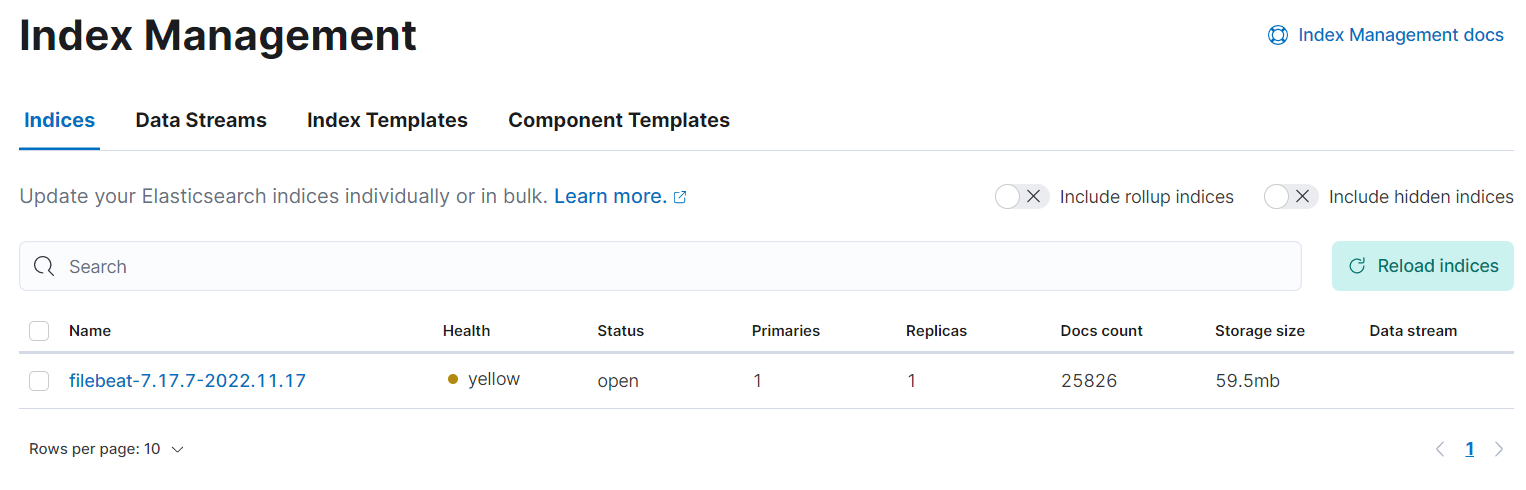
filebeat modules enable mysql

Oke giờ vào kibana -> discovery

Kiểm tra index, ta có 1 index là filebeat-7.17.7-2022.11.17

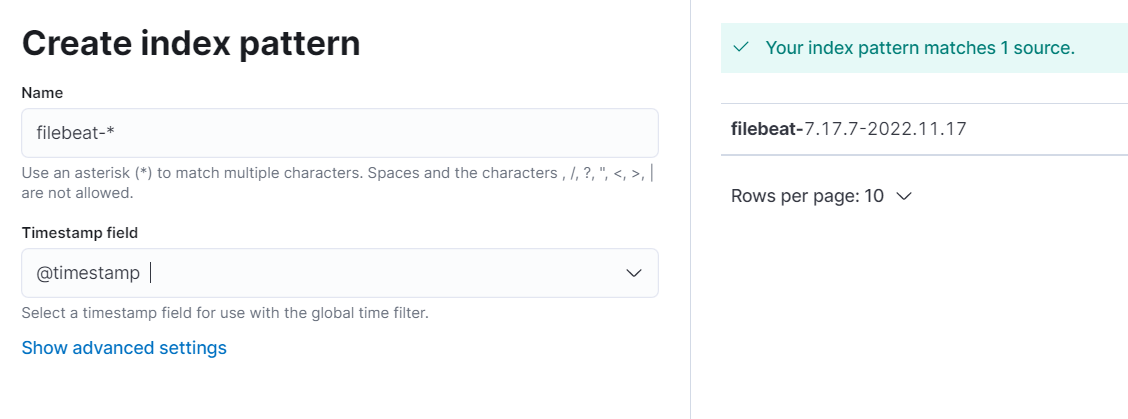
Đây chính log mà filebeat gửi đến logstash và giờ đã lưu trong ELK





Nó đang lưu được 25826 log

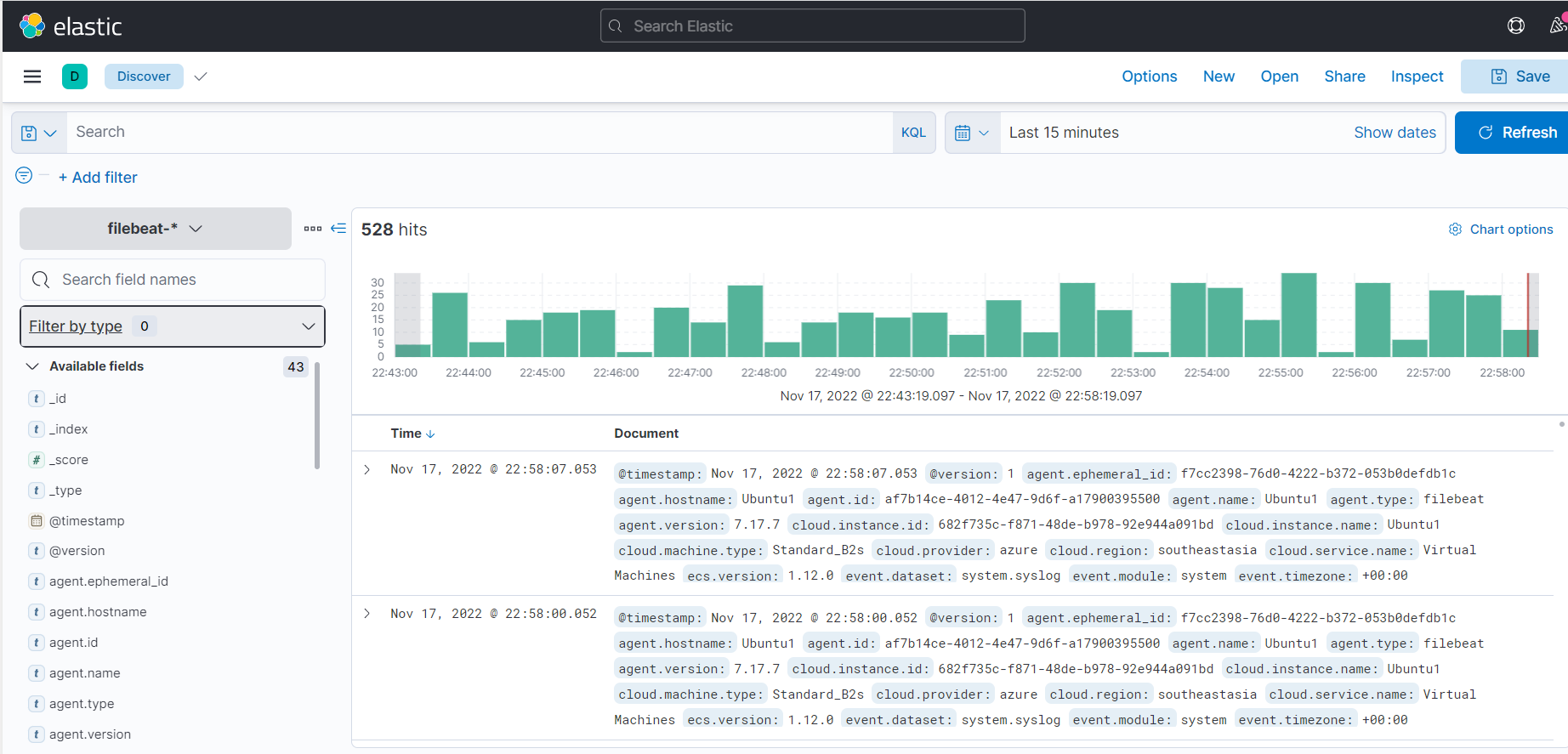
Để quan sát log này ta vào index Patterns

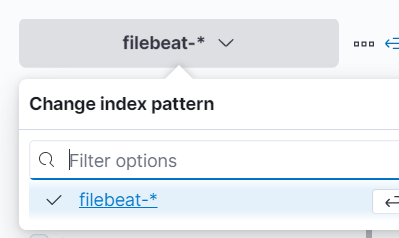


Pattern tên : filebeat-\*

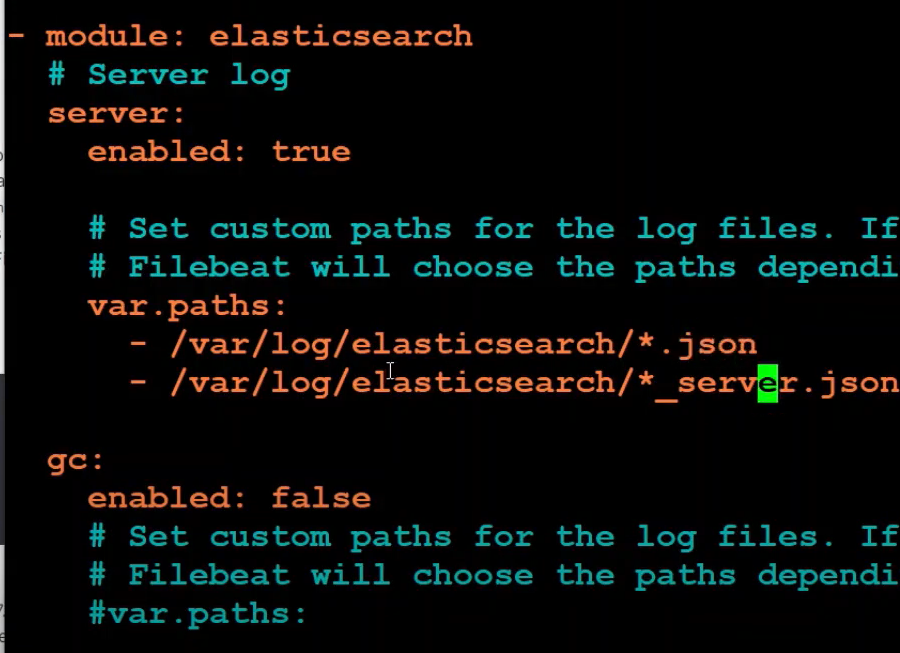
Tức là đọc tất cả index có tiền tố phù hợp vs pattern này

Giờ quay trở lại phần discover





Index có dạng là filebeat



Logs cho K8S

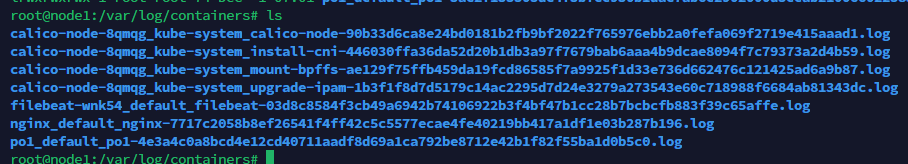
Tôi bị 1 case là deploy daemonSet filebeat để monitor mà chỉ nhận được mỗi 2 logs



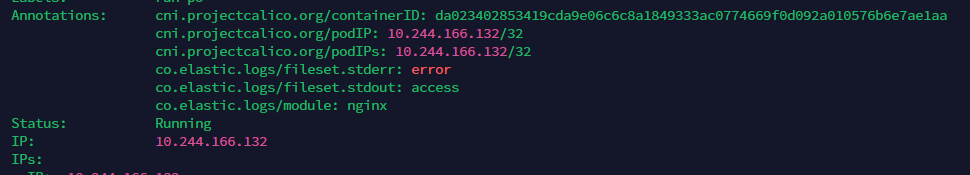
Bởi vì sao

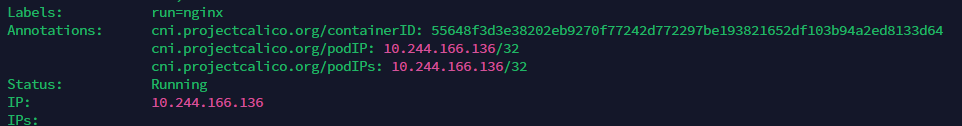
2 con pod này thường xuyên bị restart lên nó sẽ update lại annotation cho pod để thằng daemonSet kia monitor

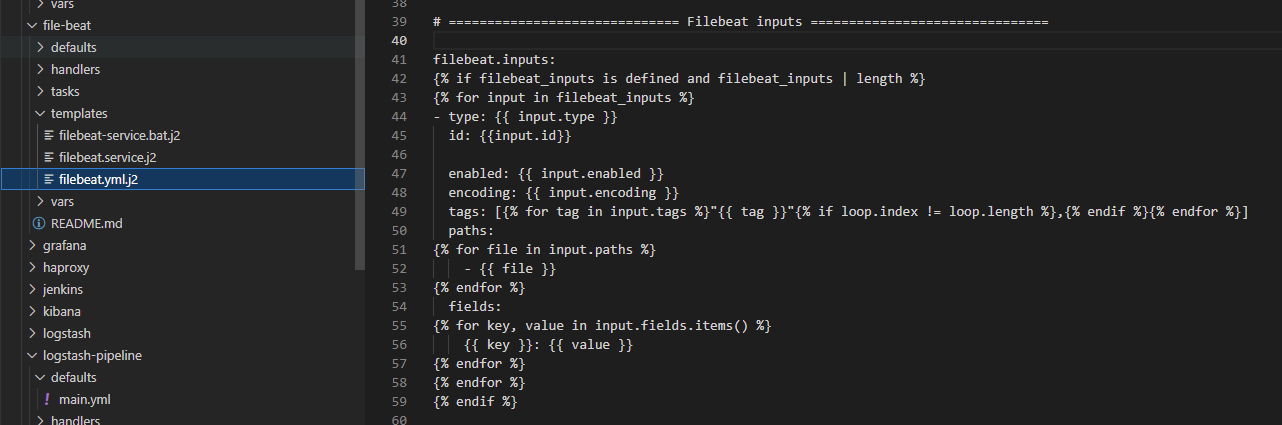
Vì vậy khi deploy daemonset xog phải restart lại hết các pod nếu n monitor được pod



Tôi restart lại những pod và nó đã được update







Filebeat này là template để cài trên window, hoặc trên các con VM khác

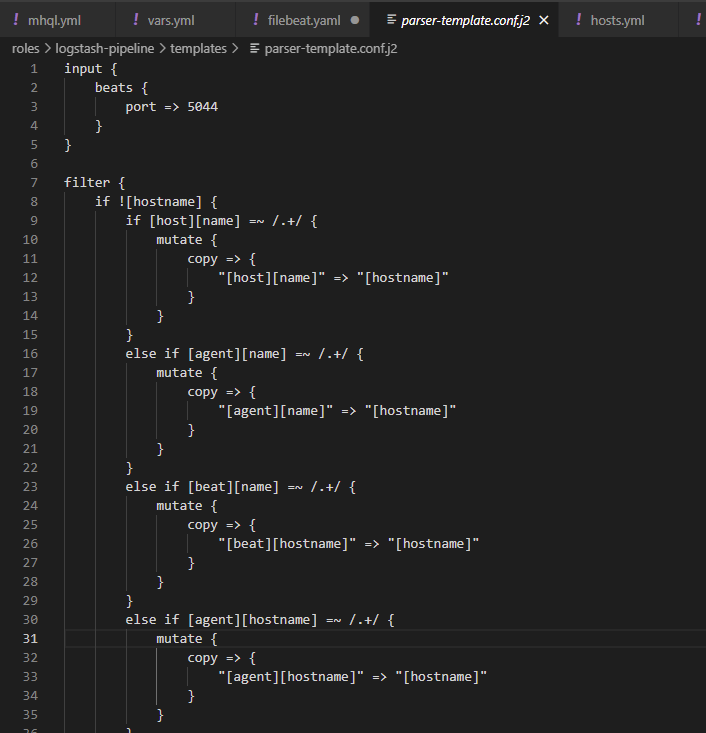
Vì vậy nếu ta có daemonSet cho filebeat rồi thì ko cần template này

Template này ta đã có trong daemonSet rồi

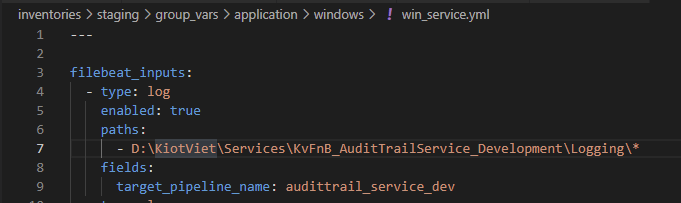
Bgio mình chỉ cần quan tâm phần xử lý log trên con ELK thôi

Logstash -> kafka -> elk

1: chỉnh phần filter đẩy đến kafka



Template này sẽ có biến là field,..

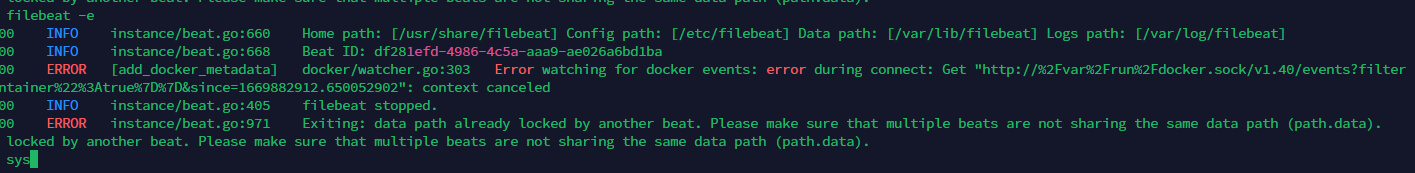


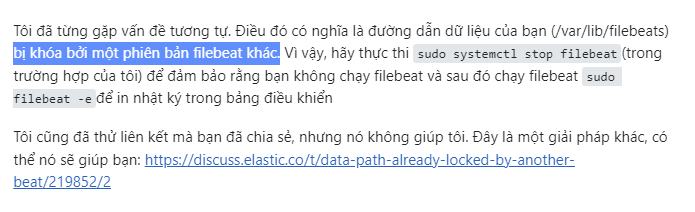
Trường field khá quan trọng để phân chia từng phần trong elk để dễ quản lý

Này để phục vụ cho logstash trên server để logstash n xử lý phần input

Đường dẫn là đường dẫn của /var/log/container/….\*.log

Đó là logs của pod lưu ở folder của node, để lấy được log này thì ta sẽ cần cài filebeat trên từng node hoặc dùng daemonSet

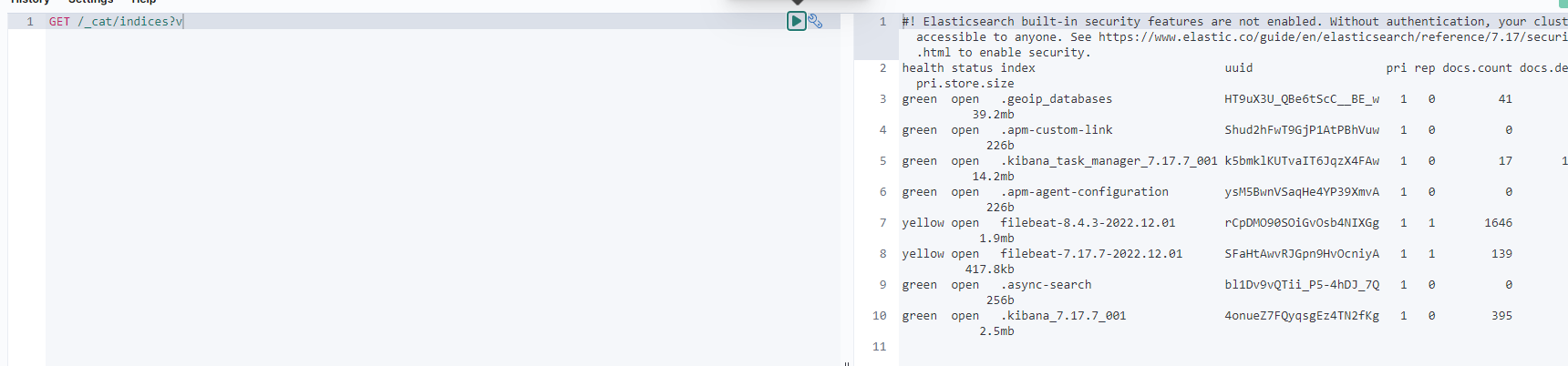




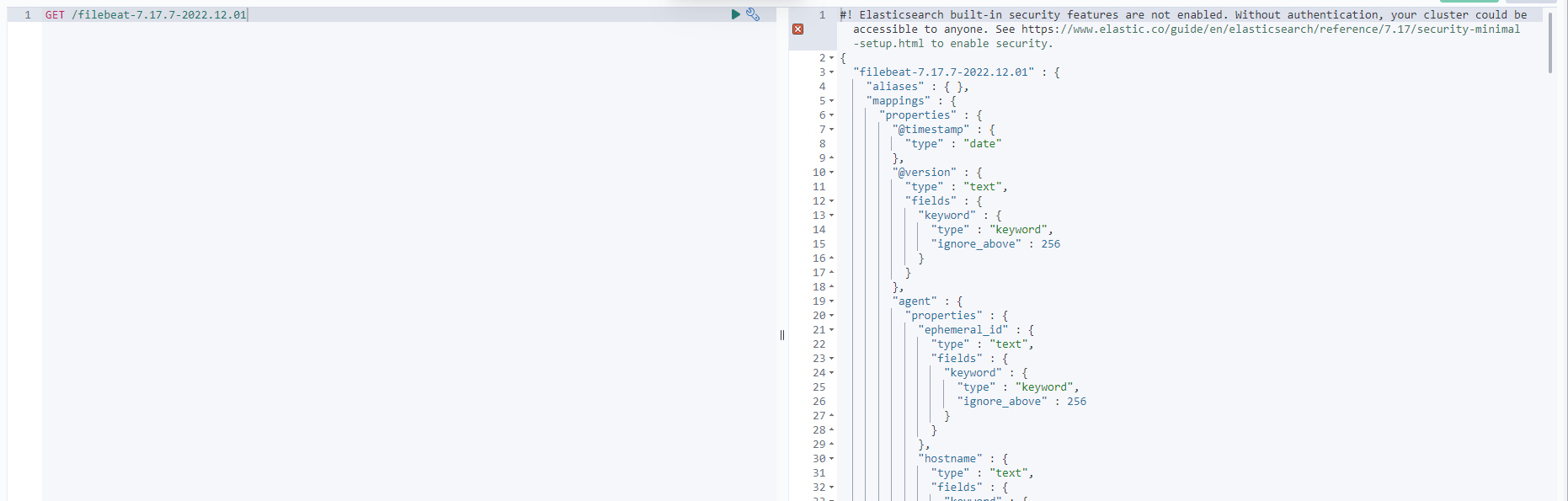
Đúng là do trước đó đã cài 1 version khác

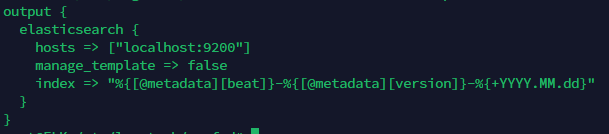
Install filebeat

GET /\_cat/indices?v



GET /filebeat-7.17.7-2022.12.01







Tạo thêm 1 file /etc/logstash/conf.d/nginx.conf

input {

  beats {

    port => 5045

  }

}

## filter config

filter {

  json {

        skip\_on\_invalid\_json => true

        source => "message"

        target => "jsonData"

        add\_tag => [ "\_message\_json\_parsed" ]

      }

  mutate {

        remove\_field => ["offset", "prospector","@version","source","host","beat","log","input","agent.hostname","cloud.machine.type","agent.name","cloud.instance.id"]

          # remove mấy thông tin này ở logs

      }

}

#send the output to the destination here it is console .

output {

    elasticsearch {

     hosts => "http://localhost:9200"

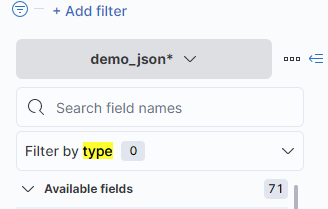
     index => "demo\_json"

  }

  stdout{}

}

Ta có đầu ra demo\_json



# Logstash: import và parse Json

