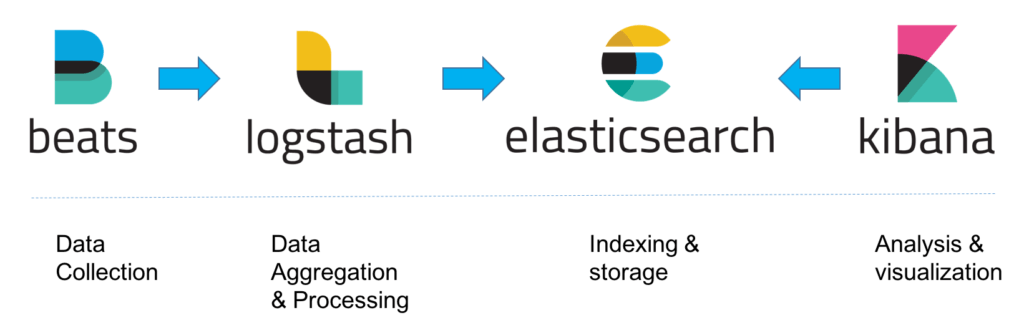
# Introduction

The ELK Stack is the world’s most popular **log management platform**.

It was a collection of three open-source products – [Elasticsearch](https://logz.io/tag/elasticsearch/), [Logstash](https://logz.io/tag/logstash/) and[Kibana](https://logz.io/tag/kibana/) – all developed by [Elastic](https://www.elastic.co/). Then the introduction of [Beats](https://www.elastic.co/downloads/beats) turned the stack into a four-legged project and led to a renaming of the stack as the *Elastic Stack*:

* **Elasticsearch**: A full-text **search engine**, based on the Apache Lucene search engine.
* **Logstash**: A **log aggregator** that collects data from various input sources, executes different transformations and enhancements, and then ships the data to supported output destinations.
* **Kibana**: A **visualization layer** that works on top of Elasticsearch, providing users with the ability to analyze and visualize the data.
* **Beats**: Lightweight agents that are installed on edge hosts to **collect different types of data** forforwarding into the stack.

Briefly, the classic architecture looks as follows:



In a more advanced architecture for handling more complex pipelines and large amounts of data in production, additional components are added into your logging architecture, for resiliency (**Kafka**, **RabbitMQ**, **Redis**) and security (**NGINX**):



Elastic Stack and Product Documentation: <https://www.elastic.co/guide/index.html>

# Installation (Windows)

First of all, download Elasticsearch, Logstash, Kibana and Beats [here](https://www.elastic.co/downloads) for Windows.

**Note**: The various shippers belonging to the Beats family. Pick the one you need. Don't have to install all. In this example, we choose Filebeat.

Elasticsearch and Logstash require **Java**. So install the latest version of JDK as Elastic recommends. Don't forget to add Java to PATH.

## Elasticsearch

After downloading, extract Elasticsearch archive into a folder of your choice.

To start Elasticsearch, run the elasticsearch.bat file from command line:

# cd to Elasticsearch folder first

.\bin\elasticsearch

Verify the server by sending a GET request to http://localhost:9200 (or you can just visit the URL from a browser):

# On cmd

curl -X GET "localhost:9200/?pretty"

# On PowerShell

# Invoke-RestMethod http://localhost:9200/?pretty

The expected output is something like this (field values will be different on different PCs):

{

"name" : "DESKTOP-RBU81GI",

"cluster\_name" : "elasticsearch",

"cluster\_uuid" : "J\_RUbtc0TB6HH6cVr4zEyQ",

"version" : {

"number" : "7.8.0",

"build\_flavor" : "default",

"build\_type" : "zip",

"build\_hash" : "757314695644ea9a1dc2fecd26d1a43856725e65",

"build\_date" : "2020-06-14T19:35:50.234439Z",

"build\_snapshot" : false,

"lucene\_version" : "8.5.1",

"minimum\_wire\_compatibility\_version" : "6.8.0",

"minimum\_index\_compatibility\_version" : "6.0.0-beta1"

},

"tagline" : "You Know, for Search"

}

**Notes**:

* Elasticsearch by default work on **port 9200**.
* Elasticsearch loads its configuration from the config\elasticsearch.yml file by default. More details: [here](https://kb.objectrocket.com/elasticsearch/how-to-configure-the-yaml-file-for-elasticsearch) and [here](https://www.elastic.co/guide/en/elasticsearch/reference/7.8/settings.html).

**Tips**:

* The postfix "?pretty" on the above URL is just for displaying data in JSON format. Removing it doesn't affect the workflow, but the output will be packed into one lengthy, hard-to-read line.
* Elasticsearch can be installed as a service to run in the background or start automatically at boot time without any user interaction. Check here for [guide](https://www.elastic.co/guide/en/elasticsearch/reference/current/zip-windows.html#windows-service).
* Elasticsearch can be installed on Windows using the .msi package. Check here for [guide](https://www.elastic.co/guide/en/elasticsearch/reference/current/windows.html).

## Kibana

After downloading, extract Kibana archive into a folder of your choice.

Modify config\kibana.yml below:

elasticsearch.hosts: ["http://localhost:9200"]

To start Kibana, run the kibana.bat file from command line:

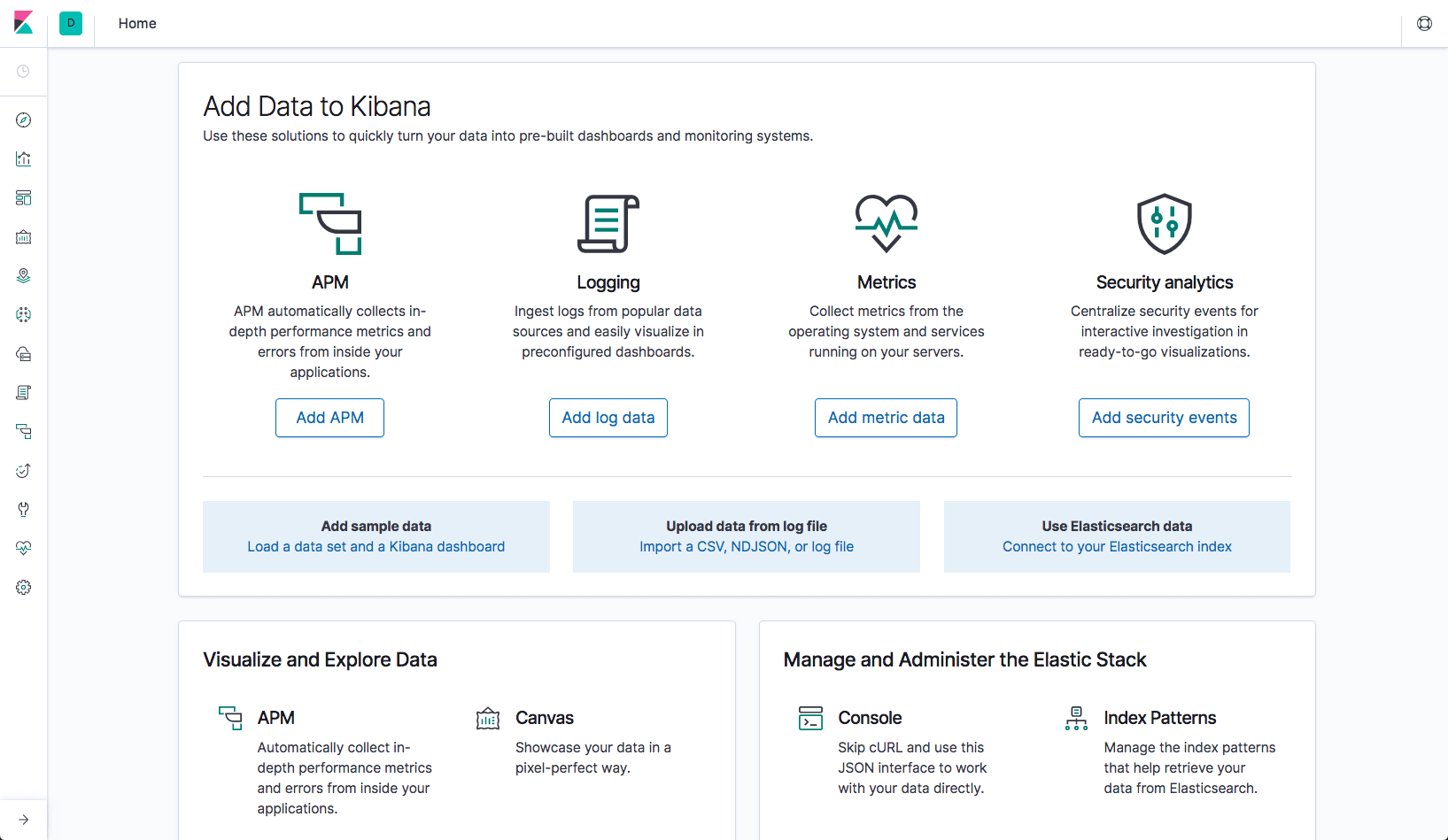
# cd to Kibana folder first

.\bin\kibana

If all proceeds as expected, the command output will display only GREEN info messages with the last message reporting:

[info][status][ui settings] Status changed from yellow to green - Ready

Verify the server by visiting http://localhost:5601. If it works, you will be presented with the Kibana home page:



**Notes**:

* Kibana by default work on **port 5601**.
* Kibana loads its configuration from the config\kibana.yml file by default. The format of this config file is explained in [Configuring Kibana](https://www.elastic.co/guide/en/kibana/7.8/settings.html).

## Logstash

After downloading, extract Logstash archive into a folder of your choice.

Prepare a config file with following content:

input { stdin { } }

output {

elasticsearch { hosts => ["localhost:9200"] }

stdout { codec => rubydebug }

}

To start Logstash with the config file, run:

# cd to Logstash folder first

.\bin\logstash -f <path-to-config-file>

To verify whether Logstash works, send any string from the command line (stdin). Logstash works when you can receive the response:

abc

{

"@version" => "1",

"@timestamp" => 2020-07-28T03:14:30.515Z,

"message" => "abc\r",

"host" => "DESKTOP-RBU81GI"

}

**Notes**:

* Logstash by default work on **port 9600**.
* Logstash loads its configuration from the config\logstash.yml file by default. The format of this config file is explained in [Configuring Kibana](https://www.elastic.co/guide/en/logstash/current/configuration.html).

## Filebeat

After downloading, extract Filebeat archive into a folder of your choice.

To install Filebeat as a service, run the .\install-service-filebeat.ps1 file from PowerShell (NOT cmd):

# cd to Filebeat folder first

.\bin\install-service-filebeat.ps1

The script might fail in case its execution is disabled. If so, set the execution policy for it:

PowerShell.exe -ExecutionPolicy UnRestricted

-File .\install-service-filebeat.ps1

Filebeat is installed as a service, but is not yet running.

Before running it, you need to configure the filebeat.yml configuration file (in the root dir of Filebeat).

Change the supplied prospector settings to track Elasticsearch logs instead of Linux logs:

- type: log

# Change to true to enable this input configuration.

enabled: false

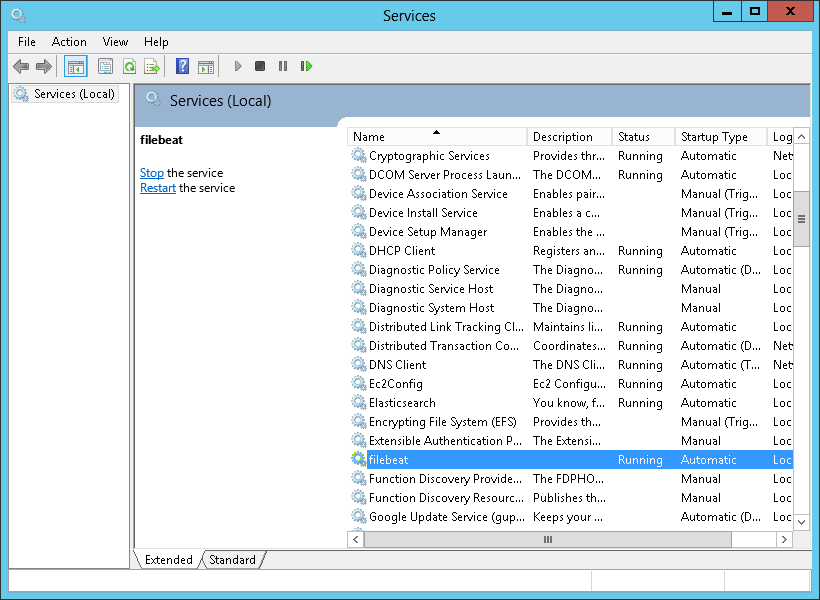
# Paths that should be crawled and fetched. Glob based paths.

paths:

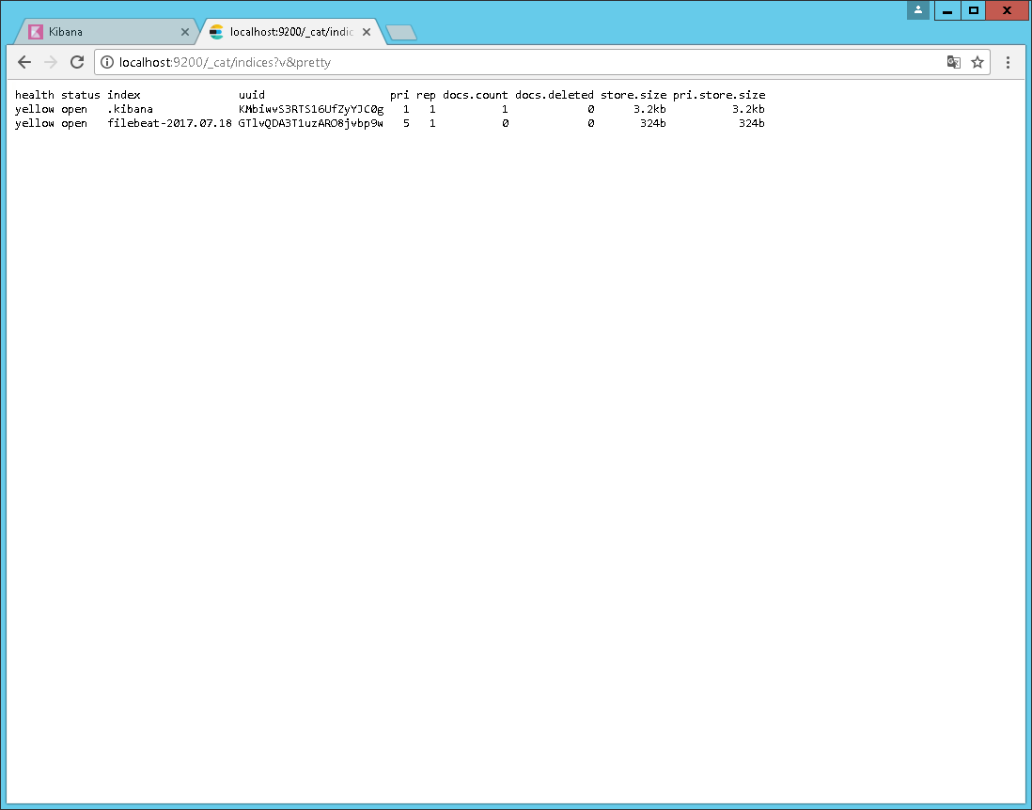
#- /var/log/\*.log

- c:\programdata\elasticsearch\logs\\*

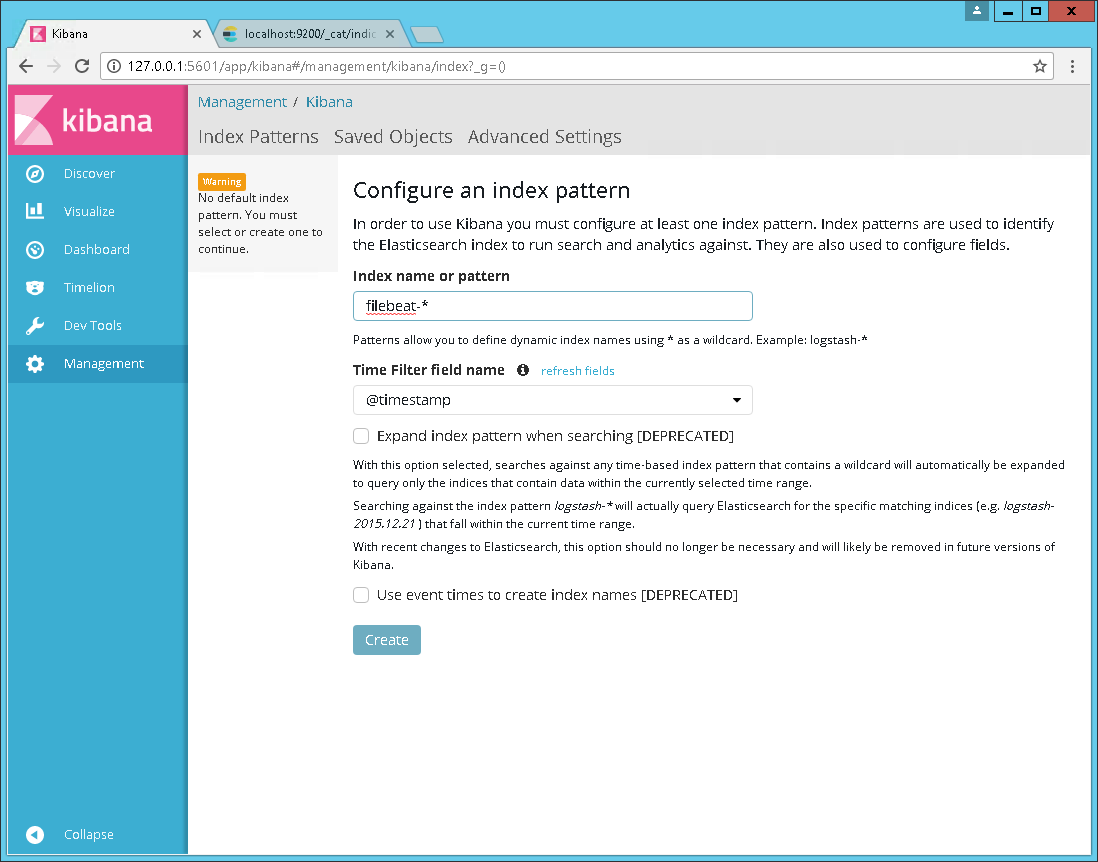
Open the Services window to start the Filebeat service:



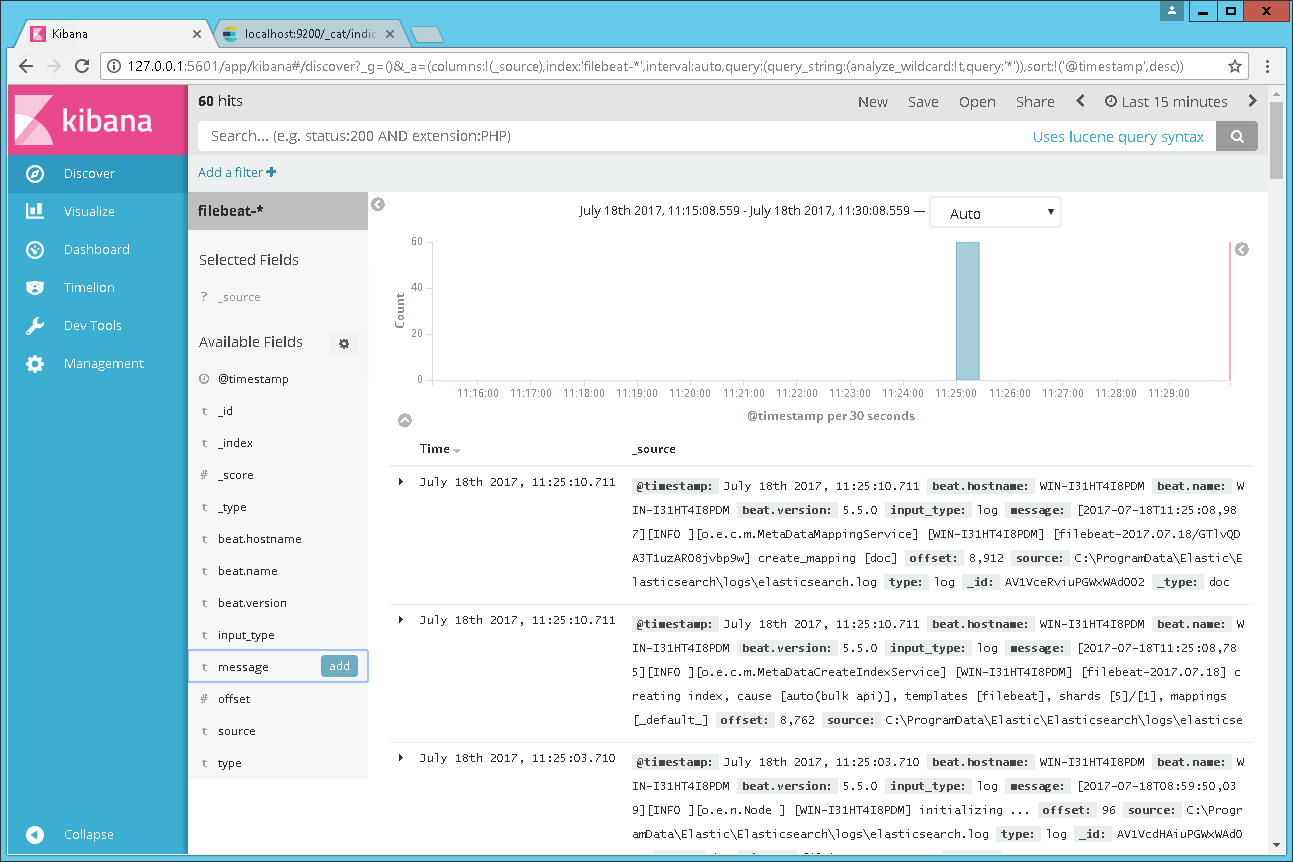
If you query Elasticsearch indices, you should see a Filebeat index created:



Back in Kibana, you can now define the filebeat-\* index pattern:



Logs belonging to the Filebeat index will then appear in the Discover page and all that’s left to do is analyze away!



<http://localhost:5601/app/kibana#/home/tutorial_directory>

References:

<https://logz.io/blog/elastic-stack-windows/>

<https://www.elastic.co/downloads/elasticsearch>

<https://www.elastic.co/downloads/kibana>

<https://www.elastic.co/downloads/logstash>

# Build Kibana Source Code

*(Tested on Ubuntu-18.04 and WSL2-Ubuntu-18.04)*

Warning:

**On Windows, it's very difficult (and maybe cannot) to build Kibana source code** because Kibana requires some packages (such as FPM, etc.) which do not support Windows. Also, there are some weird issues (due to incomparability, I guess) during the building process which I cannot resolve.

**To conclude:**

**1. Building Kibana source code on Linux is the best choice!**

**2. For Windows, must use WSL2 or Docker with a Linux container.**

Follow steps below to build source code of Kibana:

**1. Clone source code of Kibana from GitHub**

git clone <https://github.com/elastic/kibana.git>

cd kibana

git checkout v7.8.0 # This version is stable

Note: Here we use branch "v7.8.0" instead of branch Master because v7.8.0 was already released and thus much more stable.

**2. Install build dependencies**

sudo apt update

sudo apt install -y curl git g++ gzip libssl-dev make python3 tar wget patch

**3. Install NodeJS**

The version of NodeJS we're going to install must match the version described in kibana/.node-version file. Otherwise, we'll get error at step 5. For Kibana 7.8.0, we use NodeJS 10.21.0.

curl -sL https://deb.nodesource.com/setup\_10.x | sudo -E bash -

sudo apt install -y nodejs

sudo npm install -g n

sudo n 10.21.0

**4. Install Yarn**

curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -

echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /etc/apt/sources.list.d/yarn.list

sudo apt update && sudo apt install yarn

**5. Install Bootstrap and all dependencies for Kibana**

# On Kibana root directory

yarn kbn bootstrap --oss

Notes:

* This step takes quite a long time (~30 minutes).
* If you have failures during yarn kbn bootstrap, you may have some corrupted packages in your yarn cache which you can clean with: yarn cache clean

**6. Build source code of Kibana**

# On Kibana root directory

yarn build --skip-os-packages --oss

Here we add:

* Option "--skip-os-packages" excludes producing rpm/deb/docker packages.
* Option "-oss" excludes X-Pack when running command. X-Pack release contains commercially licensed code. Whereas, the OSS release is the one that contains only open-source code.

Notes:

* This step takes quite a long time (~40 minutes).
* This step does not only compile JS and TS code, but also downloads packages, copy source code, remove temporary generated files, creates snapshot, etc.
* The repo's size after building completes is large (~4.5 GB in total).
* If any dependency / package is missing during the building process, install it using yarn instead of npm.

**7. Test (Optional)**

Warning: Increase node.js heap size

Kibana is a big project and for some commands it can happen that the process hits the default heap limit and crashes with an out-of-memory error. If you run into this problem, you can increase maximum heap size by setting the --max\_old\_space\_size option on the command line. To set the limit for all commands, simply add the following line to your shell config:

export NODE\_OPTIONS="--max\_old\_space\_size=2048"

To unit test Kibana, run:

# On Kibana root directory

yarn test --force

Note: This step takes a very long time (several hours).

**8. Run Elasticsearch**

Kibana requires an Elasticsearch instance to be running. So we need to run the Elasticsearch snapshot (located in kibana\target):

yarn es snapshot --license oss

Note: Here we specify an optional license with the --license flag. There are:

* --license basic: gives access to only basic capabilities. This is the default option.
* --license trial: gives access to all capabilities.
* --license oss: gives access to open-source capabilities. We use this option here because in step 5 and 6, we chose to build OSS Kibana. If we use --license basic or trial, we have to modify the config and data generated by the Elasticsearch snapshot, which if not doing correctly, will cause errors in step 10.

Then visit <http://localhost:9200> in a browser, and enter username as "elastic" and password as "changeme" to login. Read more about [Elastic built-in users](https://www.elastic.co/guide/en/elasticsearch/reference/current/built-in-users.html#:~:text=Built%2Din%20usersedit,the%20built%2Din%20user%20passwords.).

The expected output is something like this (field values will be different on different PCs):

{

"name" : "DESKTOP-RBU81GI",

"cluster\_name" : "elasticsearch",

"cluster\_uuid" : "J\_RUbtc0TB6HH6cVr4zEyQ",

"version" : {

"number" : "7.8.0",

"build\_flavor" : "default",

"build\_type" : "zip",

"build\_hash" : "757314695644ea9a1dc2fecd26d1a43856725e65",

"build\_date" : "2020-06-14T19:35:50.234439Z",

"build\_snapshot" : false,

"lucene\_version" : "8.5.1",

"minimum\_wire\_compatibility\_version" : "6.8.0",

"minimum\_index\_compatibility\_version" : "6.0.0-beta1"

},

"tagline" : "You Know, for Search"

}

**9. Extract Kibana snapshot**

Snapshots will be generated after the building process completes. They are located in kibana/target directory. Here we will use the Linux-version snapshot to run Kibana.

sudo mkdir /usr/share/kibana/

# On Kibana root directory

sudo tar -xzf target/kibana-oss-7.8.0-SNAPSHOT-linux-x86\_64.tar.gz -C /usr/share/kibana --strip-components 1

sudo ln -sf /usr/share/kibana/bin/\* /usr/bin/

Read about more options for [Running Elasticsearch during development](https://www.elastic.co/guide/en/kibana/master/running-elasticsearch.html).

**10. Run Kibana**

Before running Kibana, we need to modify its configuration.

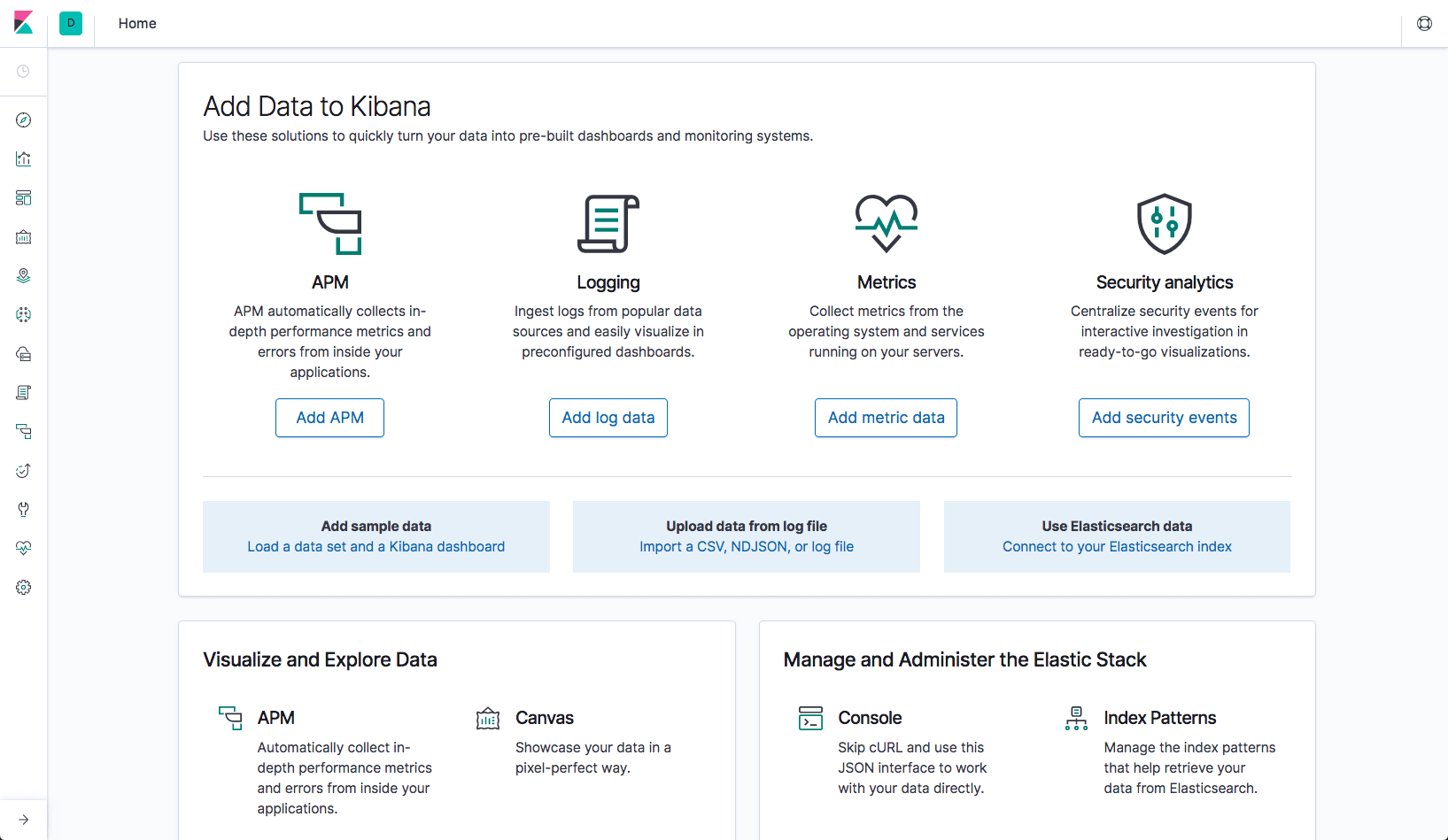
Open /usr/share/kibana/config/kibana.yml, and add these settings:

elasticsearch.hosts: ["http://localhost:9200"]

Now run Kibana:

kibana

Verify the server by visiting <http://localhost:5601>. If it works, you will be presented with the Kibana home page:



**Tips:**

* To view all commands supported by Yarn on Kibana, run:

$ yarn run

yarn run v1.22.4

…

info Project commands

- build

node scripts/build --all-platforms

- build:types

tsc --p tsconfig.types.json

- checkLicenses

node scripts/check\_licenses --dev

- debug

node --nolazy --inspect scripts/kibana --dev

- debug-break

node --nolazy --inspect-brk scripts/kibana --dev

- docs:acceptApiChanges

node --max-old-space-size=6144 scripts/check\_published\_api\_changes.js --accept

- es

node scripts/es

- kbn

node scripts/kbn

- kbn:bootstrap

yarn build:types && node scripts/register\_git\_hook

- kbn:watch

node scripts/kibana --dev --logging.json=false

- lint

yarn run lint:es && yarn run lint:sass

- lint:es

node scripts/eslint

- lint:sass

node scripts/sasslint

- makelogs

node scripts/makelogs

- preinstall

node ./preinstall\_check

- spec\_to\_console

node scripts/spec\_to\_console

- start

node scripts/kibana --dev

- storybook

node scripts/storybook

- test

grunt test

- test:coverage

grunt test:coverage

- test:ftr

node scripts/functional\_tests

- test:ftr:runner

node scripts/functional\_test\_runner

- test:ftr:server

node scripts/functional\_tests\_server

- test:jest

node scripts/jest

- test:jest\_integration

node scripts/jest\_integration

- test:karma

grunt test:karma

- test:karma:debug

grunt test:karmaDebug

- test:mocha

node scripts/mocha

- test:mocha:coverage

grunt test:mochaCoverage

- typespec

typings-tester --config x-pack/plugins/canvas/public/lib/aeroelastic/tsconfig.json x-pack/plugins/canvas/public/lib/aeroelastic/\_\_fixtures\_\_/typescript/typespec\_tests.ts

- uiFramework:build

cd packages/kbn-ui-framework && yarn docSiteBuild

- uiFramework:createComponent

cd packages/kbn-ui-framework && yarn createComponent

- uiFramework:documentComponent

cd packages/kbn-ui-framework && yarn documentComponent

- uiFramework:start

cd packages/kbn-ui-framework && yarn docSiteStart

* To view all options of **build** command, run:

$ build --all-platforms --help

usage: node scripts/build

build the Kibana distributable

options:

--oss Only produce the OSS distributable of Kibana

--no-oss Only produce the default distributable of Kibana

--skip-archives Don't produce tar/zip archives

--skip-os-packages Don't produce rpm/deb/docker packages

--all-platforms Produce archives for all platforms, not just this one

--rpm Only build the rpm package

--deb Only build the deb package

--docker Only build the docker image

--skip-docker-ubi Don't build the docker ubi image

--release Produce a release-ready distributable

--version-qualifier Suffix version with a qualifier

--skip-node-download Reuse existing downloads of node.js

--verbose,-v Turn on verbose logging

--no-debug Turn off debug logging

References:

<https://www.elastic.co/guide/en/kibana/master/development-getting-started.html>

<https://www.elastic.co/guide/en/kibana/master/building-kibana.html>

<https://github.com/linux-on-ibm-z/docs/wiki/Building-Kibana>

# Create Docker Images for Elasticsearch, Logstash and Kibana

Following steps below to create Docker images for Elasticsearch, Logstash and Kibana. These images run on an Ubuntu 18.04 container.

## Create Java Image

Create a base image named "java\_image" to pre-install a few required libraries for Elastic Stack.

We will use Ubuntu 18.04 with latest JRE and a user called "esuser" to avoid starting Elasticsearch as the root user.

The Dockerfile for java\_image:

FROM ubuntu:18.04

RUN apt update

RUN apt install -y default-jre

RUN useradd -d /home/esuser -m esuser

WORKDIR /home/esuser

ENV JAVA\_HOME /usr/lib/jvm/default-java

To create an image (with custom tag "java\_image"), run:

docker build -t java\_image .

## Create Elasticsearch Image

Before creating the Dockerfile for Elasticsearch called "es\_image", we should create an elasticsearch.yml file (let's put it in the same location as the Dockerfile):

The elasticsearch.yml file:

cluster.name: docker-cluster

node.name: docker-node-1

# Path to directory where to store the data (separate multiple locations by comma):

path.data: /home/esuser/data

# Set the bind address to a specific IP (IPv4 or IPv6):

network.host: 0.0.0.0

# Without it, error "The default discovery settings are unsuitable for production use"

discovery.seed\_hosts: []

The Dockerfile for es\_image:

FROM java\_image

ENV DEBIAN\_FRONTEND noninteractive

RUN mkdir /data

RUN apt install -y wget

RUN \

wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.8.0-linux-x86\_64.tar.gz && \

tar xvzf elasticsearch-7.8.0-linux-x86\_64.tar.gz && \

rm -f elasticsearch-7.8.0-linux-x86\_64.tar.gz && \

chown -R esuser:esuser elasticsearch-7.8.0

# elasticsearch.yml and Dockerfile are on same location

ADD elasticsearch.yml /home/esuser/elasticsearch-7.8.0/config/elasticsearch.yml

CMD elasticsearch-7.8.0/bin/elasticsearch

To create an image (with custom tag "es\_image"), run:

docker build -t es\_image .

## Create Logstash Image

Before creating the Dockerfile for Logstast called "logstash\_image", we should create a logstash.conf file (let's put it in the same location as the Dockerfile):

The logstash.conf file:

input {

file {

path => "path\to\log\file"

start\_position => "beginning"

}

}

filter {

grok {

match => { "message" => "%{COMBINEDAPACHELOG}" }

}

date {

match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]

}

geoip {

source => "clientip"

}

}

output {

elasticsearch {

hosts => ["localhost:9200"]

}

}

The Dockerfile for logstash\_image:

FROM java\_image

ENV DEBIAN\_FRONTEND noninteractive

RUN apt install -y wget

RUN \

wget https://artifacts.elastic.co/downloads/logstash/logstash-7.8.0.tar.gz && \

tar xvzf logstash-7.8.0.tar.gz && \

rm -f logstash-7.8.0.tar.gz && \

chown -R esuser:esuser logstash-7.8.0

# logstash.conf and Dockerfile are on same location

ADD logstash.conf /home/esuser

CMD logstash-7.8.0/bin/logstash -f logstash.conf --verbose

To create an image (with custom tag "logstash\_image"), run:

docker build -t logstash\_image .

## Create Kibana Image

Before creating the Dockerfile for Kibana called "kibana\_image", we should create an kibana.yml file (let's put it in the same location as the Dockerfile):

The kibana.yml file:

server.port: 5601

server.host: "0.0.0.0"

elasticsearch.hosts: ["http://localhost:9200"]

The Dockerfile for kibana\_image:

FROM java\_image

ENV DEBIAN\_FRONTEND noninteractive

RUN apt install -y wget

RUN \

wget https://artifacts.elastic.co/downloads/kibana/kibana-7.8.0-linux-x86\_64.tar.gz && \

tar xvzf kibana-7.8.0-linux-x86\_64.tar.gz && \

rm -f kibana-7.8.0-linux-x86\_64.tar.gz && \

chown -R esuser:esuser kibana-7.8.0-linux-x86\_64

ADD kibana.yml kibana-7.8.0-linux-x86\_64/config/kibana.yml

CMD kibana-7.8.0-linux-x86\_64/bin/kibana

To create an image (with custom tag "kibana\_image"), run:

docker build -t kibana\_image .

## Run Images on Container

Start Elasticsearch:

# Start a single-node Elasticsearch cluster inside a container named 'es\_dev'

# and exposes it on ports 9200 and 9300

docker run --user esuser \

--name es\_dev \

-p 9200:9200 \

-p 9300:9300 \

-e "discovery.type=single-node"

es\_image

# Check connection to Elasticsearch (JSON should be returned)

curl "http://localhost:9200"

**Note**:

* On Linux, if you get error "max virtual memory areas vm.max\_map\_count […] is too slow, increase to at least [262144]", run the following command then rebuild the Elasticsearch image:

vm.max\_map\_count = 262144

Start Kibana:

# Start Kibana inside a container named 'kibana\_dev' on port 5601

# and links it to the container (es\_dev) where the Elasticsearch cluster is running

docker run --user esuser \

--name kibana\_dev \

--link es\_dev:elasticsearch \

-p 5601:5601 \

kibana\_image

# Check connection to Kibana (HTML should be returned)

curl http://localhost:5601 --location

# Run Elasticsearch and Kibana on Docker

## Pulling Docker Images

# Elasticsearch 7.8.0

docker pull docker.elastic.co/elasticsearch/elasticsearch:7.8.0

# Kibana 7.8.0

docker pull docker.elastic.co/kibana/kibana:7.8.0

# Logstash 7.8.0

docker pull docker.elastic.co/logstash/logstash:7.8.0

Full list of Docker images of all versions for Elastic Stack: <https://www.docker.elastic.co/>

## Running Docker Images

# Start a single-node Elasticsearch cluster inside a container named 'es\_dev'

# and exposes it on ports 9200 and 9300

docker run \

--name es\_dev \

-p 9200:9200 \

-p 9300:9300 \

-e "discovery.type=single-node"

docker.elastic.co/elasticsearch/elasticsearch:7.8.0

# Start Kibana inside a container named 'kibana\_dev' on port 5601

# and links it to the container (es\_dev) where the Elasticsearch cluster is running

docker run \

--name kibana\_dev \

--link es\_dev:elasticsearch \

-p 5601:5601 \

docker.elastic.co/kibana/kibana:7.8.0

# Check connection to Elasticsearch (JSON should be returned)

curl "http://localhost:9200"

# Check connection to Kibana (HTML should be returned)

curl http://localhost:5601 --location

**Tip**:

* To stop a running container, then remove it, run:

docker container stop <container-name> && docker container rm <container-name>

## Configuring

Elasticsearch and Kibana can be configured using YAML. You can create config files and point to them when starting the containers.

* Elasticsearch: add -v path/to/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml to docker run command.
* Kibana: add -v path/to/kibana.yml:/usr/share/kibana/config/kibana.yml to docker run command.

Complete guide for running Elastic Stack on Docker:

<https://logz.io/learn/docker-monitoring-elk-stack/>

# Logstash Configuration

<https://www.elastic.co/guide/en/logstash/current/config-examples.html>

<https://qbox.io/blog/elasticsearch-logstash-kibana-apache-logs>

# Beats

All supported logs: <http://localhost:5601/app/kibana#/home/tutorial_directory>

# Elasticsearch and PHP

<https://www.elastic.co/guide/en/elasticsearch/client/php-api/current/index.html>

<https://www.tutorialspoint.com/elasticsearch/index.htm>

# Troubleshoot

Index exists but Kibana doesn't show any result in “Discover” tab:

<https://stackoverflow.com/a/31438315>

<https://stackoverflow.com/a/34129101>

# X-Pack

X-Pack is a set of features that extend the Elastic Stack, that is Elasticsearch, Kibana, Logstash and Beats. This includes features like security, monitoring, machine learning, reporting, etc.