**Setting up ELK Stack (Elastic Stack) on AWS EC2 instance**

[Avarjana Panditha](https://acpasavarjana.medium.com/?source=post_page-----fc2e1b006fe3--------------------------------)

[Avarjana Panditha](https://acpasavarjana.medium.com/?source=post_page-----fc2e1b006fe3--------------------------------)

[Sep 23, 2020·10 min read](https://acpasavarjana.medium.com/setting-up-elk-stack-elastic-stack-on-aws-ec2-instance-fc2e1b006fe3?source=post_page-----fc2e1b006fe3--------------------------------)



It’s true that AWS has its own ElasticSearch service but what if you need to future proof your deployment in case of a platform migration. The following guide is for you.

Steps

* Create an EC2 instance.
* Install Java.
* Install Elasticseach.
* Install Logstash.
* Install Kibana.

Create an EC2 instance

If you are an absolute beginner, then follow these steps to create an EC2 instance. For a production environment, it is recommended to have at least three separate instances for each module. Otherwise, you can just create an EC2 with following configs,

* Ubuntu 20.04 or 18.04 LTS (64Bit x86).
* At least m4.large.
* Storage 15GB.

We need to create an EC2 instance with **Ubuntu Server 20.04 LTS (HVM)**AMI (Amazon Machine Image). Select that from the list of AMIs. Ubuntu Server 18.04 LTS is also fine but try to stick with the latest version at the moment. Make sure the **64-bit (x86)**is selected.

https://miro.medium.com/max/60/1*ROpMOot5JjOz3tFyIQ1cbQ.png?q=20



Ubuntu Server 20.04 LTS (HVM)

In the next step, select the **m4.micro**instance type. You are free to select anything greater than that but do not go for less because that might make your life miserable with ELK.

https://miro.medium.com/max/60/1*p_GAeBqT-63bKFP3haH95A.png?q=20



m4.large instance selection

You can keep the default values at **Step 3: Configure Instance Details**and move forward to the **Add Storage**step. Make the size of the disk space as 15GB. For the record 10GB is enough but make it 15GB to have a small buffer on your hand.

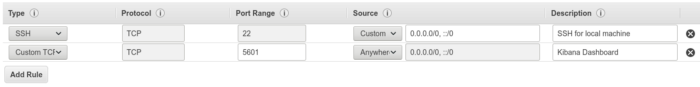
https://miro.medium.com/max/60/1*CR47X8XYqOrTXyerYmNxVQ.png?q=20



Storage Configuration

I do recommend adding some tags because it will make your life easier with AWS resources. Next up is the security group config. To access Kibana dashboard we need to expose TCP port 5601. Default SSH one can be kept as it is. If you have a static public IP for your PC, make sure to limit the source to that IP.

https://miro.medium.com/max/60/1*1gzlt4ajiGWRzxANU3XhhA.png?q=20



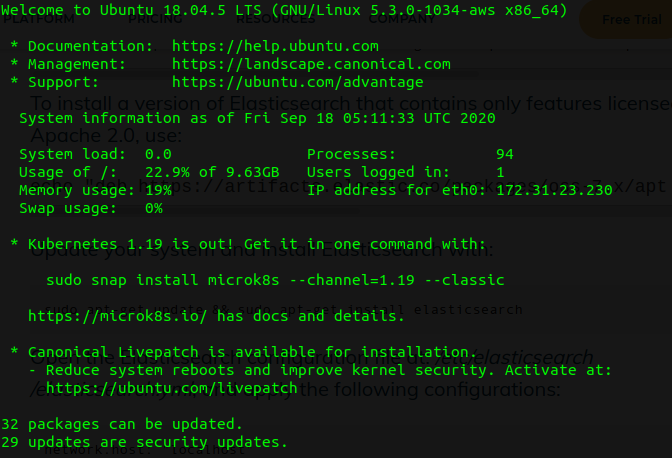
Security config

You are all set to go. Now click on the **Review and launch**button to create your instance. Make sure to keep the “pem” file somewhere safe. Now you need to SSH into the newly created instance. Go to the EC2 dashboard and click the Connect button on top after selecting your instance. From that copy the code which is in the **example**section. It will be something like,

ssh -i "<something-something>.pem" ubuntu@ec2-\*\*-\*\*-\*-\*\*\*.compute-1.amazonaws.com

Now you need to open a terminal if your local machine is Linux based or if you have a windows machine, I prefer the **Git Bash** because it will be easier to use with SSH. You can also use Putty or similar. Make sure to open the terminal in the location where you have your “pem” file. Now paste the copied SSH command and run it. You will be presented with something like,

https://miro.medium.com/max/60/1*d1K55mtY0474OOqcx8OSqA.png?q=20



Successfully SSH into the EC2 instance

As a good practice just execute **sudo apt-get update**once you successfully SSH into the EC2 instance.

Install Java

Now we need to install Java. It is quite a straightforward installation. Just execute the command to get **default-jre.**

sudo apt-get install default-jre

You can always check your Java version by executing the command **java -version**to make sure you have Java installed successfully.

Install Elasticsearch

Let’s install Elasticsearch as the first step towards completing our ELK stack. The installation process is fairly straightforward. Follow the next few steps. Add the repository key first,

wget -qO - <https://artifacts.elastic.co/GPG-KEY-elasticsearch> | sudo apt-key add -

Install the Apt Transport HTTPS because the default image does not contain that.

sudo apt-get install apt-transport-https

Now run the following commands to complete the installation.

echo "deb <https://artifacts.elastic.co/packages/7.x/apt> stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.listsudo apt-get update && sudo apt-get install elasticsearch

You have successfully installed Elasticsearch on your EC2 instance but we are not done yet. We need to check the config file now. Open the file, **“*/etc/elasticsearch/elasticsearch.yml”***in your preferred editor (vim, nano). I recommend you to do this after you switch to superuser by executing **“sudo su”**command. Now you need to edit the config file as follows. The lines which are in bold letters are the changes to be done.

# ======================== Elasticsearch Configuration =========================  
#  
# NOTE: Elasticsearch comes with reasonable defaults for most settings.  
# Before you set out to tweak and tune the configuration, make sure you  
# understand what are you trying to accomplish and the consequences.  
#  
# The primary way of configuring a node is via this file. This template lists  
# the most important settings you may want to configure for a production cluster.  
#  
# Please consult the documentation for further information on configuration options:  
# <https://www.elastic.co/guide/en/elasticsearch/reference/index.html>  
#  
# ---------------------------------- Cluster -----------------------------------  
#  
# Use a descriptive name for your cluster:  
#  
#cluster.name: my-application  
#  
# ------------------------------------ Node ------------------------------------  
#  
# Use a descriptive name for the node:  
#  
**node.name: node-1**  
#  
# Add custom attributes to the node:  
#  
#node.attr.rack: r1  
#  
# ----------------------------------- Paths ------------------------------------  
#  
# Path to directory where to store the data (separate multiple locations by comma):  
#  
path.data: /var/lib/elasticsearch  
#  
# Path to log files:  
#  
path.logs: /var/log/elasticsearch  
#  
# ----------------------------------- Memory -----------------------------------  
#  
# Lock the memory on startup:  
#  
#bootstrap.memory\_lock: true  
#  
# Make sure that the heap size is set to about half the memory available  
# on the system and that the owner of the process is allowed to use this  
# limit.  
#  
# Elasticsearch performs poorly when the system is swapping the memory.  
#  
# ---------------------------------- Network -----------------------------------  
#  
# Set the bind address to a specific IP (IPv4 or IPv6):  
#  
**network.host: "localhost"**  
#  
# Set a custom port for HTTP:  
#  
**http.port: 9200**  
#  
# For more information, consult the network module documentation.  
#  
# --------------------------------- Discovery ----------------------------------  
#  
# Pass an initial list of hosts to perform discovery when this node is started:  
# The default list of hosts is ["127.0.0.1", "[::1]"]  
#  
#discovery.seed\_hosts: ["host1", "host2"]  
#  
# Bootstrap the cluster using an initial set of master-eligible nodes:  
#  
**cluster.initial\_master\_nodes: ["node-1"]**  
#  
# For more information, consult the discovery and cluster formation module documentation.  
#  
# ---------------------------------- Gateway -----------------------------------  
#  
# Block initial recovery after a full cluster restart until N nodes are started:  
#  
#gateway.recover\_after\_nodes: 3  
#  
# For more information, consult the gateway module documentation.  
#  
# ---------------------------------- Various -----------------------------------  
#  
# Require explicit names when deleting indices:  
#  
#action.destructive\_requires\_name: true

You need to configure JVM heap size to be in the safe side when running the ElasticSearch. Since we are running an m4.large instance, we will allocate 4GB as the heap size. Edit the file **“/etc/elasticsearch/jvm.options”**and enable the “-Xms4g” lines. Refer the screenshot below,

https://miro.medium.com/max/60/1*IuaPWDSRFUiFozrwHb9R5A.png?q=20



Enable 4GB heap for Elasticsearch

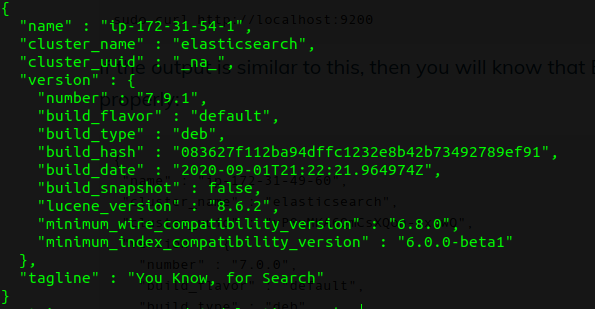
Now you are ready to start the Elasticseach service by executing, (If you have not switched to superuser by running “sudo su”, then please run these commands with sudo prefix)

service elasticsearch start

After running the service, you can verify that your installation is working perfectly by executing a CURL command,

curl http://localhost:9200

https://miro.medium.com/max/60/1*1nR57L224BSMI-VPSEcghg.png?q=20



Output of the “curl http://localhost:9200”

Install Logstash

Hey, you are almost there. Now moving on to next letter in the ELK stack. Just execute this command to install Logstash.

sudo apt-get install logstash

Now create a sample file at the location “/home” named as “test\_log.log” and include just a few lines for testing. Some content like,

INFO somelog  
ERR somelog2  
ERR somelog3

We need to create a config file for Logstash to test the installation. Create a file with **“sudo nano /etc/logstash/conf.d/apache-01.conf”**and include the following content into the file.

input {  
 file {  
 path => "/home/test\_log.log"  
 start\_position => "beginning"  
 }  
}output{  
 elasticsearch {  
 hosts => ["localhost:9200"]  
 }  
}

You can start the Logstash service by running,

sudo service logstash start

And check if the config so far is running without error by executing this CURL command.

sudo curl -XGET 'localhost:9200/\_cat/indices?v&pretty'

It should provide an output like,

https://miro.medium.com/max/60/1*en7wgeUeJg7byyQ7mdm6iQ.png?q=20

https://miro.medium.com/max/700/1*en7wgeUeJg7byyQ7mdm6iQ.png

CURL command output of indices

Install Kibana

The last and most important part of the installation process. Execute the following command to install the Kibana dashboard.

sudo apt-get install kibana

Now we need to configure Kibana to match our Elasticsearch search port and our EC2 opened port. Edit the config file using a preferred editor.

/etc/kibana/kibana.yml

Make sure the bold lines in the following config is changed.

# Kibana is served by a back end server. This setting specifies the port to use.  
**server.port: 5601**# Specifies the address to which the Kibana server will bind. IP addresses and host names are both valid values.  
# The default is 'localhost', which usually means remote machines will not be able to connect.  
# To allow connections from remote users, set this parameter to a non-loopback address.  
**server.host: "0.0.0.0"**# Enables you to specify a path to mount Kibana at if you are running behind a proxy.  
# Use the `server.rewriteBasePath` setting to tell Kibana if it should remove the basePath  
# from requests it receives, and to prevent a deprecation warning at startup.  
# This setting cannot end in a slash.  
#server.basePath: ""# Specifies whether Kibana should rewrite requests that are prefixed with  
# `server.basePath` or require that they are rewritten by your reverse proxy.  
# This setting was effectively always `false` before Kibana 6.3 and will  
# default to `true` starting in Kibana 7.0.  
#server.rewriteBasePath: false# The maximum payload size in bytes for incoming server requests.  
#server.maxPayloadBytes: 1048576# The Kibana server's name. This is used for display purposes.  
#server.name: "your-hostname"# The URLs of the Elasticsearch instances to use for all your queries.  
**elasticsearch.hosts: ["**[**http://localhost:9200**](http://localhost:9200/)**"]**# When this setting's value is true Kibana uses the hostname specified in the server.host  
# setting. When the value of this setting is false, Kibana uses the hostname of the host  
# that connects to this Kibana instance.  
#elasticsearch.preserveHost: true# Kibana uses an index in Elasticsearch to store saved searches, visualizations and  
# dashboards. Kibana creates a new index if the index doesn't already exist.  
#kibana.index: ".kibana"# The default application to load.  
#kibana.defaultAppId: "home"# If your Elasticsearch is protected with basic authentication, these settings provide  
# the username and password that the Kibana server uses to perform maintenance on the Kibana  
# index at startup. Your Kibana users still need to authenticate with Elasticsearch, which  
# is proxied through the Kibana server.  
#elasticsearch.username: "kibana\_system"  
#elasticsearch.password: "pass"# Enables SSL and paths to the PEM-format SSL certificate and SSL key files, respectively.  
# These settings enable SSL for outgoing requests from the Kibana server to the browser.  
#server.ssl.enabled: false  
#server.ssl.certificate: /path/to/your/server.crt  
#server.ssl.key: /path/to/your/server.key# Optional settings that provide the paths to the PEM-format SSL certificate and key files.  
# These files are used to verify the identity of Kibana to Elasticsearch and are required when  
# xpack.security.http.ssl.client\_authentication in Elasticsearch is set to required.  
#elasticsearch.ssl.certificate: /path/to/your/client.crt  
#elasticsearch.ssl.key: /path/to/your/client.key# Optional setting that enables you to specify a path to the PEM file for the certificate  
# authority for your Elasticsearch instance.  
#elasticsearch.ssl.certificateAuthorities: [ "/path/to/your/CA.pem" ]# To disregard the validity of SSL certificates, change this setting's value to 'none'.  
#elasticsearch.ssl.verificationMode: full# Time in milliseconds to wait for Elasticsearch to respond to pings. Defaults to the value of  
# the elasticsearch.requestTimeout setting.  
#elasticsearch.pingTimeout: 1500# Time in milliseconds to wait for responses from the back end or Elasticsearch. This value  
# must be a positive integer.  
#elasticsearch.requestTimeout: 30000# List of Kibana client-side headers to send to Elasticsearch. To send \*no\* client-side  
# headers, set this value to [] (an empty list).  
#elasticsearch.requestHeadersWhitelist: [ authorization ]# Header names and values that are sent to Elasticsearch. Any custom headers cannot be overwritten  
# by client-side headers, regardless of the elasticsearch.requestHeadersWhitelist configuration.  
#elasticsearch.customHeaders: {}# Time in milliseconds for Elasticsearch to wait for responses from shards. Set to 0 to disable.  
#elasticsearch.shardTimeout: 30000# Time in milliseconds to wait for Elasticsearch at Kibana startup before retrying.  
#elasticsearch.startupTimeout: 5000# Logs queries sent to Elasticsearch. Requires logging.verbose set to true.  
#elasticsearch.logQueries: false# Specifies the path where Kibana creates the process ID file.  
#pid.file: /var/run/kibana.pid# Enables you to specify a file where Kibana stores log output.  
#logging.dest: stdout# Set the value of this setting to true to suppress all logging output.  
#logging.silent: false# Set the value of this setting to true to suppress all logging output other than error messages.  
#logging.quiet: false# Set the value of this setting to true to log all events, including system usage information  
# and all requests.  
#logging.verbose: false# Set the interval in milliseconds to sample system and process performance  
# metrics. Minimum is 100ms. Defaults to 5000.  
#ops.interval: 5000# Specifies locale to be used for all localizable strings, dates and number formats.  
# Supported languages are the following: English - en , by default , Chinese - zh-CN .  
#i18n.locale: "en"

Now start the Kibana service.

sudo service kibana start

Wait for like 30–45 seconds and visit your EC2 instance’s public IP with port 5601.

http://<IP>:5601

You will be redirected to Kibana dashboard home.

That’s all you have to do to have a running ELK stack on top of an AWS EC2 instance. Feel free to ask questions in the comments section if anything is blurry. Happy logging :D

TIP: Make sure to terminate all the EC2 instances when not in use for more than one day.