**Requirements**

**Host setup**

* [Docker Engine](https://docs.docker.com/get-docker/) version **18.06.0** or newer
* [Docker Compose](https://docs.docker.com/compose/install/) version **1.26.0** or newer (including [Compose V2](https://docs.docker.com/compose/cli-command/))
* 1.5 GB of RAM

*ℹ️ Especially on Linux, make sure your user has the*[*required permissions*](https://docs.docker.com/engine/install/linux-postinstall/)*to interact with the Docker daemon.*

By default, the stack exposes the following ports:

* 5044: Logstash Beats input
* 5000: Logstash TCP input
* 9600: Logstash monitoring API
* 9200: Elasticsearch HTTP
* 9300: Elasticsearch TCP transport
* 5601: Kibana

**Elasticsearch's**[**bootstrap checks**](https://www.elastic.co/guide/en/elasticsearch/reference/current/bootstrap-checks.html)**were purposely disabled to facilitate the setup of the Elastic stack in development environments. For production setups, we recommend users to set up their host according to the instructions from the Elasticsearch documentation:**[**Important System Configuration**](https://www.elastic.co/guide/en/elasticsearch/reference/current/system-config.html)**.**

**Docker Desktop**

**Windows**

If you are using the legacy Hyper-V mode of *Docker Desktop for Windows*, ensure [File Sharing](https://docs.docker.com/desktop/windows/#file-sharing) is enabled for the C: drive.

**macOS**

The default configuration of *Docker Desktop for Mac* allows mounting files from /Users/, /Volume/, /private/, /tmp and /var/folders exclusively. Make sure the repository is cloned in one of those locations or follow the instructions from the [documentation](https://docs.docker.com/desktop/mac/#file-sharing) to add more locations.

**Usage**

**⚠️ You must rebuild the stack images with docker-compose build whenever you switch branch or update the**[**version**](https://github.com/deviantony/docker-elk#version-selection)**of an already existing stack.**

**Bringing up the stack**

Clone this repository onto the Docker host that will run the stack, then start the stack's services locally using Docker Compose:

$ docker-compose up

*ℹ️ You can also run all services in the background (detached mode) by appending the -d flag to the above command.*

Give Kibana about a minute to initialize, then access the Kibana web UI by opening [http://localhost:5601](http://localhost:5601/) in a web browser and use the following (default) credentials to log in:

* user: *elastic*
* password: *changeme*

*Upon the initial startup, the elastic, logstash\_internal and kibana\_system Elasticsearch users are intialized with the values of the passwords defined in the*[*.env*](https://github.com/deviantony/docker-elk/blob/main/.env)*file ("changeme" by default). The first one is the*[*built-in superuser*](https://www.elastic.co/guide/en/elasticsearch/reference/current/built-in-users.html)*, the other two are used by Kibana and Logstash respectively to communicate with Elasticsearch. This task is only performed during the initial startup of the stack. To change users' passwords after they have been initialized, please refer to the instructions in the next section.*

**Initial setup**

**Setting up user authentication**

*Refer to*[*Security settings in Elasticsearch*](https://www.elastic.co/guide/en/elasticsearch/reference/current/security-settings.html)*to disable authentication.*

**Starting with Elastic v8.0.0, it is no longer possible to run Kibana using the bootstraped privileged elastic user.**

The *"changeme"* password set by default for all aforementioned users is **unsecure**. For increased security, we will reset the passwords of all aforementioned Elasticsearch users to random secrets.

1. Reset passwords for default users

The commands below resets the passwords of the elastic, logstash\_internal and kibana\_system users. Take note of them.

$ docker-compose exec elasticsearch bin/elasticsearch-reset-password --batch --user elastic

$ docker-compose exec elasticsearch bin/elasticsearch-reset-password --batch --user logstash\_internal

$ docker-compose exec elasticsearch bin/elasticsearch-reset-password --batch --user kibana\_system

If the need for it arises (e.g. if you want to [collect monitoring information](https://www.elastic.co/guide/en/logstash/current/monitoring-with-metricbeat.html) through Beats and other components), feel free to repeat this operation at any time for the rest of the [built-in users](https://www.elastic.co/guide/en/elasticsearch/reference/current/built-in-users.html).

1. Replace usernames and passwords in configuration files

Replace the password of the elastic user inside the .env file with the password generated in the previous step. Its value isn't used by any core component, but [extensions](https://github.com/deviantony/docker-elk#how-to-enable-the-provided-extensions) use it to connect to Elasticsearch.

*ℹ️ In case you don't plan on using any of the provided*[*extensions*](https://github.com/deviantony/docker-elk#how-to-enable-the-provided-extensions)*, or prefer to create your own roles and users to authenticate these services, it is safe to remove the ELASTIC\_PASSWORD entry from the .env file altogether after the stack has been initialized.*

Replace the password of the logstash\_internal user inside the .env file with the password generated in the previous step. Its value is referenced inside the Logstash pipeline file (logstash/pipeline/logstash.conf).

Replace the password of the kibana\_system user inside the .env file with the password generated in the previous step. Its value is referenced inside the Kibana configuration file (kibana/config/kibana.yml).

See the [Configuration](https://github.com/deviantony/docker-elk#configuration) section below for more information about these configuration files.

1. Restart Logstash and Kibana to re-connect to Elasticsearch using the new passwords

$ docker-compose up -d logstash kibana

*Learn more about the security of the Elastic stack at*[*Secure the Elastic Stack*](https://www.elastic.co/guide/en/elasticsearch/reference/current/secure-cluster.html)*.*

**Injecting data**

Open the Kibana web UI by opening [http://localhost:5601](http://localhost:5601/) in a web browser and use the following credentials to log in:

* user: *elastic*
* password: *<your generated elastic password>*

Now that the stack is fully configured, you can go ahead and inject some log entries. The shipped Logstash configuration allows you to send content via TCP:

# Using BSD netcat (Debian, Ubuntu, MacOS system, ...)

$ cat /path/to/logfile.log | nc -q0 localhost 5000

# Using GNU netcat (CentOS, Fedora, MacOS Homebrew, ...)

$ cat /path/to/logfile.log | nc -c localhost 5000

You can also load the sample data provided by your Kibana installation.

**Cleanup**

Elasticsearch data is persisted inside a volume by default.

In order to entirely shutdown the stack and remove all persisted data, use the following Docker Compose command:

$ docker-compose down -v

**Version selection**

This repository stays aligned with the latest version of the Elastic stack. The main branch tracks the current major version (8.x).

To use a different version of the core Elastic components, simply change the version number inside the [.env](https://github.com/deviantony/docker-elk/blob/main/.env) file. If you are upgrading an existing stack, remember to rebuild all container images using the docker-compose build command.

**Always pay attention to the**[**official upgrade instructions**](https://www.elastic.co/guide/en/elasticsearch/reference/current/setup-upgrade.html)**for each individual component before performing a stack upgrade.**

Older major versions are also supported on separate branches:

* [release-7.x](https://github.com/deviantony/docker-elk/tree/release-7.x): 7.x series
* [release-6.x](https://github.com/deviantony/docker-elk/tree/release-6.x): 6.x series (End-of-life)
* [release-5.x](https://github.com/deviantony/docker-elk/tree/release-5.x): 5.x series (End-of-life)

**Configuration**

*Configuration is not dynamically reloaded, you will need to restart individual components after any configuration change.*

**How to configure Elasticsearch**

The Elasticsearch configuration is stored in [elasticsearch/config/elasticsearch.yml](https://github.com/deviantony/docker-elk/blob/main/elasticsearch/config/elasticsearch.yml).

You can also specify the options you want to override by setting environment variables inside the Compose file:

elasticsearch:

environment:

network.host: \_non\_loopback\_

cluster.name: my-cluster

Please refer to the following documentation page for more details about how to configure Elasticsearch inside Docker containers: [Install Elasticsearch with Docker](https://www.elastic.co/guide/en/elasticsearch/reference/current/docker.html).

**How to configure Kibana**

The Kibana default configuration is stored in [kibana/config/kibana.yml](https://github.com/deviantony/docker-elk/blob/main/kibana/config/kibana.yml).

You can also specify the options you want to override by setting environment variables inside the Compose file:

kibana:

environment:

SERVER\_NAME: kibana.example.org

Please refer to the following documentation page for more details about how to configure Kibana inside Docker containers: [Install Kibana with Docker](https://www.elastic.co/guide/en/kibana/current/docker.html).

**How to configure Logstash**

The Logstash configuration is stored in [logstash/config/logstash.yml](https://github.com/deviantony/docker-elk/blob/main/logstash/config/logstash.yml).

You can also specify the options you want to override by setting environment variables inside the Compose file:

logstash:

environment:

LOG\_LEVEL: debug

Please refer to the following documentation page for more details about how to configure Logstash inside Docker containers: [Configuring Logstash for Docker](https://www.elastic.co/guide/en/logstash/current/docker-config.html).

**How to disable paid features**

Switch the value of Elasticsearch's xpack.license.self\_generated.type setting from trial to basic (see [License settings](https://www.elastic.co/guide/en/elasticsearch/reference/current/license-settings.html)).

You can also cancel an ongoing trial before its expiry date — and thus revert to a basic license — either from the [License Management](https://www.elastic.co/guide/en/kibana/current/managing-licenses.html) panel of Kibana, or using Elasticsearch's [Licensing APIs](https://www.elastic.co/guide/en/elasticsearch/reference/current/licensing-apis.html).

**How to scale out the Elasticsearch cluster**

Follow the instructions from the Wiki: [Scaling out Elasticsearch](https://github.com/deviantony/docker-elk/wiki/Elasticsearch-cluster)

**How to reset a password programmatically**

If for any reason your are unable to use Kibana to change the password of your users (including [built-in users](https://www.elastic.co/guide/en/elasticsearch/reference/current/built-in-users.html)), you can use the Elasticsearch API instead and achieve the same result.

In the example below, we reset the password of the elastic user (notice "/user/elastic" in the URL):

$ curl -XPOST -D- 'http://localhost:9200/\_security/user/elastic/\_password' \

-H 'Content-Type: application/json' \

-u elastic:<your current elastic password> \

-d '{"password" : "<your new password>"}'

**Extensibility**

**How to add plugins**

To add plugins to any ELK component you have to:

1. Add a RUN statement to the corresponding Dockerfile (eg. RUN logstash-plugin install logstash-filter-json)
2. Add the associated plugin code configuration to the service configuration (eg. Logstash input/output)
3. Rebuild the images using the docker-compose build command

**How to enable the provided extensions**

A few extensions are available inside the [extensions](https://github.com/deviantony/docker-elk/blob/main/extensions) directory. These extensions provide features which are not part of the standard Elastic stack, but can be used to enrich it with extra integrations.

The documentation for these extensions is provided inside each individual subdirectory, on a per-extension basis. Some of them require manual changes to the default ELK configuration.

**JVM tuning**

**How to specify the amount of memory used by a service**

By default, both Elasticsearch and Logstash start with [1/4 of the total host memory](https://docs.oracle.com/javase/8/docs/technotes/guides/vm/gctuning/parallel.html#default_heap_size) allocated to the JVM Heap Size.

The startup scripts for Elasticsearch and Logstash can append extra JVM options from the value of an environment variable, allowing the user to adjust the amount of memory that can be used by each component:

| **Service** | **Environment variable** |
| --- | --- |
| Elasticsearch | ES\_JAVA\_OPTS |
| Logstash | LS\_JAVA\_OPTS |

To accomodate environments where memory is scarce (Docker for Mac has only 2 GB available by default), the Heap Size allocation is capped by default to 256MB per service in the docker-compose.yml file. If you want to override the default JVM configuration, edit the matching environment variable(s) in the docker-compose.yml file.

For example, to increase the maximum JVM Heap Size for Logstash:

logstash:

environment:

LS\_JAVA\_OPTS: -Xmx1g -Xms1g

**How to enable a remote JMX connection to a service**

As for the Java Heap memory (see above), you can specify JVM options to enable JMX and map the JMX port on the Docker host.

Update the {ES,LS}\_JAVA\_OPTS environment variable with the following content (I've mapped the JMX service on the port 18080, you can change that). Do not forget to update the -Djava.rmi.server.hostname option with the IP address of your Docker host (replace **DOCKER\_HOST\_IP**):

logstash:

environment:

LS\_JAVA\_OPTS: -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management