Docker Elastic stack

# Details

## Docker

Each service run on its own container and ports are mapped for network access.

Each service has a volume mounted to store persistent data.

The Elastic stack is entirely setup through the docker compose file.

# Host setup

## Host specifications

|  |  |
| --- | --- |
| Operating System | Ubuntu LTS |
| CPU | 2 |
| RAM | 16 Go |

## Package requirements

|  |  |
| --- | --- |
| Docker-engine | Ubuntu LTS |
| Docker-compose | 2 |

### Docker Engine install

Setup the repository

sudo apt-get update

sudo apt-get install \

ca-certificates \

curl \

gnupg \

lsb-release

Add Docker’s official GPG key:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

Set up the stable repository

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \

(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

Install Docker Engine

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io

Check the version

sudo docker run hello-world

### Docker Compose install

Install the latest stable version

sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

Apply execute right ont the binary

sudo chmod +x /usr/local/bin/docker-compose

Check the version

docker-compose --version

## Setting requirements

### vm.max\_map\_count

vm.max\_map\_count must be set to at least 262144

To do it permanently edit the value in the file **/etc/sysctl.conf** and set it to

vm.max\_map\_count=262144

### Disable swapping

Disabling swapping greatly increases Elasticsearch’s performances and stability

sudo swapoff -a

sudo rm /swap.img

To disable swapping permanently, edit the file **/etc/fstab** and comment any line containing the word *swap*

/dev/disk/by-id/dm-uuid-LVM-wa82yffOij01TO2sGjci1952xNB6LcuDTqOuEInmYOWQRCvhEnMHYsS7tQ6nkOMn / ext4 defaults 0 1

# /boot was on /dev/sda2 during curtin installation

/dev/disk/by-uuid/4d5f4028-2612-4fdc-bae3-bc6cc1231fe3 /boot ext4 defaults 0 1

#/swap.img none swap sw 0 0

# Docker config

## Config directory

Create a directory which will contain the docker compose yaml and the subdirectories for Elasticsearch, Kibana and Logstash

mkdir -p /docker/elasticsearch

mkdir /docker/kibana

mkdir /docker/Logstash

cd /docker

## Config files

Make the env file containing variables used by the docker compose yaml

echo "ELASTIC\_PASSWORD=ChangeMe$

KIBANA\_PASSWORD= ChangeMe$

STACK\_VERSION=8.1.2

ES\_PORT=9200

KIBANA\_PORT=5601

# Increase or decrease based on the available host memory (in bytes)

MEM\_LIMIT=8589934592

# Project namespace (defaults to the current folder name if not set)

COMPOSE\_PROJECT\_NAME=elastic

" > test

Set the docker-compose.yml

version: "2.2"

services:

setup:

image: docker.elastic.co/elasticsearch/elasticsearch:${STACK\_VERSION}

volumes:

- certs:/usr/share/elasticsearch/config/certs

user: "0"

command: >

bash -c '

if [ x${ELASTIC\_PASSWORD} == x ]; then

echo "Set the ELASTIC\_PASSWORD environment variable in the .env file";

exit 1;

elif [ x${KIBANA\_PASSWORD} == x ]; then

echo "Set the KIBANA\_PASSWORD environment variable in the .env file";

exit 1;

fi;

if [ ! -f certs/ca.zip ]; then

echo "Creating CA";

bin/elasticsearch-certutil ca --silent --pem -out config/certs/ca.zip;

unzip config/certs/ca.zip -d config/certs;

fi;

if [ ! -f certs/certs.zip ]; then

echo "Creating certs";

echo -ne \

"instances:\n"\

" - name: es-01\n"\

" dns:\n"\

" - es-01\n"\

" - localhost\n"\

" ip:\n"\

" - 127.0.0.1\n"\

" - name: kibana\n"\

" dns:\n"\

" - kibana\n"\

" - localhost\n"\

" ip:\n"\

" - 192.168.1.125\n"\

> config/certs/instances.yml;

bin/elasticsearch-certutil cert --silent --pem -out config/certs/certs.zip --in config/certs/instances.yml --ca-cert config/certs/ca/ca.crt --ca-key config/certs/ca/ca.key;

unzip config/certs/certs.zip -d config/certs;

fi;

echo "Setting file permissions"

chown -R root:root config/certs;

find . -type d -exec chmod 750 \{\} \;;

find . -type f -exec chmod 640 \{\} \;;

echo "Waiting for Elasticsearch availability";

until curl -s --cacert config/certs/ca/ca.crt https://es-01:9200 | grep -q "missing authentication credentials"; do sleep 30; done;

echo "Setting kibana\_system password";

until curl -s -X POST --cacert config/certs/ca/ca.crt -u elastic:${ELASTIC\_PASSWORD} -H "Content-Type: application/json" https://es-01:9200/\_security/user/kibana\_system/\_password -d "{\"password\":\"${KIBANA\_PASSWORD}\"}" | grep -q "^{}"; do sleep 10; done;

curl -s -XGET --cacert config/certs/ca/ca.crt -u elastic:${ELASTIC\_PASSWORD} https://es-01:9200|grep cluster

echo "All done!";

'

healthcheck:

test: ["CMD-SHELL", "[ -f config/certs/es-01/es-01.crt ]"]

interval: 1s

timeout: 5s

retries: 120

es-01:

depends\_on:

setup:

condition: service\_healthy

image: docker.elastic.co/elasticsearch/elasticsearch:${STACK\_VERSION}

volumes:

- certs:/usr/share/elasticsearch/config/certs

- esdata01:/usr/share/elasticsearch/data

- es-01\_logs:/var/log/elasticsearch

- ./elasticsearch/es-01.yml:/usr/share/elasticsearch/config/elasticsearch.yml

ports:

- ${ES\_PORT}:9200

environment:

- ELASTIC\_PASSWORD=${ELASTIC\_PASSWORD}

mem\_limit: ${MEM\_LIMIT}

ulimits:

memlock:

soft: -1

hard: -1

healthcheck:

test:

[

"CMD-SHELL",

"curl -s --cacert config/certs/ca/ca.crt https://localhost:9200 | grep -q 'missing authentication credentials'",

]

interval: 10s

timeout: 10s

retries: 120

kibana:

depends\_on:

es-01:

condition: service\_healthy

image: docker.elastic.co/kibana/kibana:${STACK\_VERSION}

volumes:

- certs:/usr/share/kibana/config/certs

- kibanadata:/usr/share/kibana/data

- kibana\_logs:/usr/share/kibana/logs

- ./kibana/kibana.yml:/usr/share/kibana/config/kibana.yml

ports:

- ${KIBANA\_PORT}:5601

mem\_limit: ${MEM\_LIMIT}

volumes:

certs:

driver: local

esdata01:

driver: local

kibanadata:

driver: local

es-01\_logs:

driver: local

kibana\_logs:

driver: local

The DNS fields or the host file must be set, otherwise the curl requests will fail.

# Manage the containers

All commands must be run inside the directory which contains the docker compose file

## Start the stack

Docker-compose up -d

The first execution will download the images and then initialize the stack. The next execution will directly start the containers without re-downloading the images and initializing the stack.

## Stop the stack

Docker-compose down

It will stop the containers without destroying the data stored on the mounted volumes.

## Destroy the stack

Docker-compose down -v

Using this command will destroy all data on the mounted volumes.