

Deploying ELK Stack On

**Docker Container project source code**









DONe By :Jevardhan p

# docker-compose.logs.yml





version: '3.5'



# will contain all elasticsearch data. volumes: filebeat-data:

services:

# Docker Logs Shipper

- filebeat: image: docker.elastic.co/beats/filebeat:${ELK\_VERSION} restart: always

# -e flag to log to stderr and disable syslog/file output command: -e -- strict.perms=false user: root environment:

ELASTIC\_USERNAME: ${ELASTIC\_USERNAME} ELASTIC\_PASSWORD: ${ELASTIC\_PASSWORD} KIBANA\_HOST\_PORT: ${KIBANA\_HOST}:${KIBANA\_PORT} ELASTICSEARCH\_HOST\_PORT:

https://${ELASTICSEARCH\_HOST}:${ELASTICSEARCH\_PORT}

volumes:

-

./filebeat/filebeat.docker.logs.yml:/usr/share/filebeat/ filebeat.yml:ro

-

/var/lib/docker/containers:/var/lib/docker/containers:ro

* /var/run/docker.sock:/var/run/docker.sock:ro



* filebeat-data:/var/lib/filebeat/data





# docker-compose.monitor.yml





version: '3.5' services:



# Prometheus Exporters elasticsearch-exporter:

image: justwatch/elasticsearch\_exporter:1.1.0 restart: always command: ["--es.uri",

"https://${ELASTIC\_USERNAME}:${ELASTIC\_PASSWORD}@${ELAST ICSEARCH\_HOST}:${ELASTICSEARCH\_PORT}",

"--es.ssl-skip-verify",

"--es.all",

"--es.snapshots",

"--es.indices"] ports:

- "9114:9114"

logstash-exporter: image: alxrem/prometheus-logstash-exporter restart: always ports:

* "9304:9304" command: ["-logstash.host", "${LOGSTASH\_HOST}"]

# Cluster Logs Shipper







filebeat-cluster-logs: image:

docker.elastic.co/beats/filebeat:${ELK\_VERSION} restart: always

# -e flag to log to stderr and disable syslog/file output command: -e -- strict.perms=false user: root environment:

ELASTIC\_USERNAME: ${ELASTIC\_USERNAME} ELASTIC\_PASSWORD: ${ELASTIC\_PASSWORD} KIBANA\_HOST\_PORT: ${KIBANA\_HOST}:${KIBANA\_PORT} ELASTICSEARCH\_HOST\_PORT:

https://${ELASTICSEARCH\_HOST}:${ELASTICSEARCH\_PORT}

volumes:

-

./filebeat/filebeat.monitoring.yml:/usr/share/filebeat/f ilebeat.yml:ro

-

/var/lib/docker/containers:/var/lib/docker/containers:ro

- /var/run/docker.sock:/var/run/docker.sock:ro

# docker-compose.nodes.yml





version: '3.5'

# will contain all elasticsearch data.

volumes:

elasticsearch-data-1: elasticsearch-data- 2: services: elasticsearch-1:



image: elastdocker/elasticsearch:${ELK\_VERSION} build:

args:

context: elasticsearch/

ELK\_VERSION: ${ELK\_VERSION}

restart: unless-stopped environment: ELASTIC\_USERNAME: ${ELASTIC\_USERNAME} ELASTIC\_PASSWORD: ${ELASTIC\_PASSWORD} ELASTIC\_CLUSTER\_NAME: ${ELASTIC\_CLUSTER\_NAME}

ELASTIC\_NODE\_NAME: ${ELASTIC\_NODE\_NAME\_1} ELASTIC\_INIT\_MASTER\_NODE:

${ELASTIC\_INIT\_MASTER\_NODE} ELASTIC\_DISCOVERY\_SEEDS:

${ELASTIC\_DISCOVERY\_SEEDS}

ES\_JAVA\_OPTS: -Xmx${ELASTICSEARCH\_HEAP} - Xms${ELASTICSEARCH\_HEAP} -

Des.enforce.bootstrap.checks=true bootstrap.memory\_lock: "true" volumes:

- elasticsearch-data- 1:/usr/share/elasticsearch/data

-

./elasticsearch/config/elasticsearch.yml:/usr/share/elas ticsearch/config/elasticsearch.yml

-



./elasticsearch/config/log4j2.properties:/usr/share/elas ticsearch/config/log4j2.properties secrets:



* source: elasticsearch.keystore target:

/usr/share/elasticsearch/config/elasticsearch.keystore

* source: elastic.ca target:

/usr/share/elasticsearch/config/certs/ca.crt

* source: elasticsearch.certificate target:

/usr/share/elasticsearch/config/certs/elasticsearch.crt

* source: elasticsearch.key target:

/usr/share/elasticsearch/config/certs/elasticsearch.key ulimits: memlock: soft: -1 hard:

-1 nofile:

soft: 200000

hard: 200000 elasticsearch-2:

image: elastdocker/elasticsearch:${ELK\_VERSION} build:

args:

context: elasticsearch/

ELK\_VERSION: ${ELK\_VERSION}

restart: unless-stopped environment: ELASTIC\_USERNAME: ${ELASTIC\_USERNAME} ELASTIC\_PASSWORD: ${ELASTIC\_PASSWORD} ELASTIC\_CLUSTER\_NAME: ${ELASTIC\_CLUSTER\_NAME} ELASTIC\_NODE\_NAME: ${ELASTIC\_NODE\_NAME\_2}



ELASTIC\_INIT\_MASTER\_NODE:

${ELASTIC\_INIT\_MASTER\_NODE}



ELASTIC\_DISCOVERY\_SEEDS:

${ELASTIC\_DISCOVERY\_SEEDS}

ES\_JAVA\_OPTS: -Xmx${ELASTICSEARCH\_HEAP} - Xms${ELASTICSEARCH\_HEAP} -

Des.enforce.bootstrap.checks=true bootstrap.memory\_lock: "true" volumes:

* elasticsearch-data- 2:/usr/share/elasticsearch/data

-

./elasticsearch/config/elasticsearch.yml:/usr/share/elas ticsearch/config/elasticsearch.yml

-

./elasticsearch/config/log4j2.properties:/usr/share/elas ticsearch/config/log4j2.properties secrets:

* source: elasticsearch.keystore target:

/usr/share/elasticsearch/config/elasticsearch.keystore

* source: elastic.ca target:

/usr/share/elasticsearch/config/certs/ca.crt

* source: elasticsearch.certificate target:

/usr/share/elasticsearch/config/certs/elasticsearch.crt

* source: elasticsearch.key target:



/usr/share/elasticsearch/config/certs/elasticsearch.key ulimits:



memlock:

soft: -1

hard: -1

nofile:

soft: 200000

hard: 200000



# docker-compose.setup.yml

version: '3.5' services:

keystore:

image: elastdocker/elasticsearch:${ELK\_VERSION} build:

context: elasticsearch/

args:

ELK\_VERSION: ${ELK\_VERSION}

command: bash /setup/setup-keystore.sh user: "0" volumes:

- ./secrets:/secrets

- ./setup/:/setup/ environment:

ELASTIC\_PASSWORD: ${ELASTIC\_PASSWORD}

certs:

image: elastdocker/elasticsearch:${ELK\_VERSION} build:

context: elasticsearch/

args:



ELK\_VERSION: ${ELK\_VERSION}

command: bash /setup/setup-certs.sh

user: "0" volumes:

* ./secrets:/secrets
* ./setup/:/setup



# docker-compose.tools.yml

version: '3.5' services:

rubban:

image: sherifabdlnaby/rubban:latest restart: unless-stopped

environment:

RUBBAN\_KIBANA\_HOST: "https://${KIBANA\_HOST}:${KIBANA\_PORT}"

RUBBAN\_KIBANA\_USER: ${ELASTIC\_USERNAME} RUBBAN\_KIBANA\_PASSWORD: ${ELASTIC\_PASSWORD} RUBBAN\_REFRESHINDEXPATTERN\_ENABLED: 'true' RUBBAN\_REFRESHINDEXPATTERN\_SCHEDULE: '\*/5 \* \* \* \*' RUBBAN\_REFRESHINDEXPATTERN\_PATTERNS: '\*' RUBBAN\_AUTOINDEXPATTERN\_ENABLED: 'true' RUBBAN\_AUTOINDEXPATTERN\_SCHEDULE: '\*/5 \* \* \* \*' RUBBAN\_AUTOINDEXPATTERN\_GENERALPATTERNS:

'[{"pattern":"filebeat?","timeFieldName":"@timestamp"},{ "pattern":"logstash?","timeFieldName":"@timestamp"}]'



# Dockerfile

ARG ELK\_VERSION

# https://github.com/elastic/elasticsearch-docker FROM docker.elastic.co/elasticsearch/elasticsearch:${ELK\_VERS ION}

# Add healthcheck

COPY scripts/docker-healthcheck .

HEALTHCHECK CMD sh ./docker-healthcheck

# Add your elasticsearch plugins setup here

# Example: RUN elasticsearch-plugin install analysis-icu #RUN elasticsearch-plugin install --batch repository-s3





***Filebeat.monitoring.yml***

name: filebeat-elk-monitoring filebeat.config:

modules:

path: ${path.config}/modules.d/\*.yml reload.enabled: false

#================================ Autodiscover

=======================================

# Autodiscover all containers with elasticsearch images, and add an separate input for

# each container and log type.

filebeat.autodiscover:



providers: - type: docker templates: - condition: contains:



config:

docker.container.image: elasticsearch

- module: elasticsearch

server: input:

type: container

paths: '/var/lib/docker/containers/${data.docker.container.id}/

\*.log' gc: input:

type: container

paths:

'/var/lib/docker/containers/${data.docker.container.id}/

\*.log' audit: input:

type: container

paths:

'/var/lib/docker/containers/${data.docker.container.id}/



\*.log' slowlog:



input:

type: container



paths: '/var/lib/docker/containers/${data.docker.container.id}/





\*.log' deprecation: input:



paths:

type: container

'/var/lib/docker/containers/${data.docker.container.id}/

\*.log'

* type: docker templates: - condition: contains:

docker.container.image: kibana

config:

* module: kibana log:

input:

type: container

paths:

'/var/lib/docker/containers/${data.docker.container.id}/

\*.log'

* type: docker templates: - condition: contains:



docker.container.image: logstash



config:

* module: logstash log:

input:



type: container

paths:





'/var/lib/docker/containers/${data.docker.container.id}/



\*.log' slowlog: input:

type: container

paths:

'/var/lib/docker/containers/${data.docker.container.id}/

\*.log'

processors:

- add\_cloud\_metadata: ~

# Output to ES directly. output.elasticsearch: hosts: '${ELASTICSEARCH\_HOST\_PORT}'

username: '${ELASTIC\_USERNAME}' password: '${ELASTIC\_PASSWORD}' ssl:

verification\_mode: "none"

#=================================== Kibana

==========================================

# Enable setting up Kibana

# Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.

# This requires a Kibana endpoint configuration. setup: kibana:



host: '${KIBANA\_HOST\_PORT}'

username: '${ELASTIC\_USERNAME}'





password: '${ELASTIC\_PASSWORD}'



#==================================== Monitoring

=====================================

# Enable Monitoring Beats

# Filebeat can export internal metrics to a central Elasticsearch monitoring

# cluster. This requires xpack monitoring to be enabled in Elasticsearch

# Use deprecated option to avoid current UX bug in 7.3.0 where filebeat creates a

# standalone monitoring cluster in the monitoring UI. # see: https://github.com/elastic/beats/pull/13182 xpack.monitoring: enabled: true

# elasticsearch:

# hosts: '${ELASTICSEARCH\_HOST\_PORT}'

# username: '${ELASTIC\_USERNAME}' # password: '${ELASTIC\_PASSWORD}'

#monitoring:

# enabled: true # elasticsearch:

# hosts: '${ELASTICSEARCH\_HOST\_PORT}'



# username: '${ELASTIC\_USERNAME}' # password: '${ELASTIC\_PASSWORD}' # ssl.enabled: true



# ssl.verification\_mode: none



#================================ HTTP Endpoint

======================================

# Enabled so we can monitor filebeat using filebeat exporter if needed.

# Each beat can expose internal metrics through a HTTP endpoint. For security

# reasons the endpoint is disabled by default. This feature is currently experimental. # Stats can be access through http://localhost:5066/stats . For pretty JSON output

# append ?pretty to the URL.

# Defines if the HTTP endpoint is enabled. http.enabled: true

http.host: 0.0.0.0

http.port: 5066





# Kibana.yml

## Default Kibana configuration from Kibana base image. ##

https://github.com/elastic/kibana/blob/master/src/dev/bu ild/tasks/os\_packages/docker\_generator/templates/kibana\_ yml.template.js

# server.name: kibana server.host: "0.0.0.0" # Elasticsearch Connection elasticsearch.hosts: [ "${ELASTICSEARCH\_HOST\_PORT}" ]

# SSL settings





server.ssl.enabled: true server.ssl.certificate:



/certs/kibana.crt server.ssl.key: /certs/kibana.key server.ssl.certificateAuthorities: [ "/certs/ca.crt" ] xpack.security.encryptionKey: C1tHnfrlfxSPxPlQ8BlgPB5qMNRtg5V5 xpack.encryptedSavedObjects.encryptionKey: D12GTfrlfxSPxPlGRBlgPB5qM5GOPDV5 xpack.reporting.encryptionKey:

RSCueeHKzrqzOVTJhkjt17EMnzM96LlN

## X-Pack security credentials elasticsearch.serviceAccountToken: "${KIBANA\_SERVICE\_ACCOUNT\_TOKEN}"

elasticsearch.ssl.certificateAuthorities: [ "/certs/ca.crt" ]

## Misc elasticsearch.requestTimeout: 90000

## ElastAlert Plugin

#elastalert-kibana-plugin.serverHost: elastalert #elastalert-kibana-plugin.serverPort: 3030



# Dockerfile

ARG ELK\_VERSION

# https://github.com/elastic/kibana-docker

FROM docker.elastic.co/kibana/kibana:${ELK\_VERSION}

ARG ELK\_VERSION

# Add your kibana plugins setup here

# Example: RUN kibana-plugin install <name|url>

-





# Logstash.yml

http.host: "0.0.0.0"

## X-Pack security credentials xpack.monitoring.elasticsearch.hosts:

${ELASTICSEARCH\_HOST\_PORT} xpack.monitoring.enabled: true xpack.monitoring.elasticsearch.username:

${ELASTIC\_USERNAME}

xpack.monitoring.elasticsearch.password:

${ELASTIC\_PASSWORD}

xpack.monitoring.elasticsearch.ssl.certificate\_authority

: /certs/ca.crt

***Pipelines.yml***

pipeline.id: main

path.config:

"/usr/share/logstash/pipeline/main.conf" queue.type:

memory

***Main.conf***

input { beats {





port => 5044

}

}

filter {

} output {

elasticsearch {

=> "${ELASTICSEARCH\_HOST\_PORT}"

=> "${ELASTIC\_USERNAME}"

hosts

user password =>

"${ELASTIC\_PASSWORD}" ssl => true

ssl\_certificate\_verification => false cacert => "/certs/ca.crt"

}

}

# Setup-certs.sh



set -e



OUTPUT\_DIR=/secrets/certs ZIP\_CA\_FILE=$OUTPUT\_DIR/ca.zip ZIP\_FILE=$OUTPUT\_DIR/certs.zip





printf "======= Generating Elastic Stack Certificates





=======\n"



printf "=====================================================\n

" if ! command -v unzip &>/dev/null; then printf "Installing Necessary Tools... \n" yum install -y -q -e 0 unzip; fi printf "Clearing Old Certificates if exits... \n" mkdir -p

$OUTPUT\_DIR find $OUTPUT\_DIR -type d -exec rm -rf

-- {} + mkdir -p $OUTPUT\_DIR/ca

printf "Generating CA Certificates... \n" PASSWORD=`openssl rand -base64 32`

/usr/share/elasticsearch/bin/elasticsearch-certutil ca - pass "$PASSWORD" --pem --out $ZIP\_CA\_FILE &> /dev/null printf "Generating Certificates... \n" unzip -qq

$ZIP\_CA\_FILE -d $OUTPUT\_DIR;

/usr/share/elasticsearch/bin/elasticsearch-certutil cert

--silent --pem --ca-cert $OUTPUT\_DIR/ca/ca.crt --ca-key

$OUTPUT\_DIR/ca/ca.key --ca-pass "$PASSWORD" --in

/setup/instances.yml -out $ZIP\_FILE &> /dev/null printf "Unzipping Certifications...

\n" unzip -qq $ZIP\_FILE -d $OUTPUT\_DIR; printf "Applying Permissions...

\n"



chown -R 1000:0 $OUTPUT\_DIR

set -e



GENERATED\_KEYSTORE=/usr/share/elasticsearch/config/elast icsearch.keystore OUTPUT\_KEYSTORE=/secrets/keystore/elasticsearch.keystore GENERATED\_SERVICE\_TOKENS=/usr/share/elasticsearch/config

/service\_tokens OUTPUT\_SERVICE\_TOKENS=/secrets/service\_tokens OUTPUT\_KIBANA\_TOKEN=/secrets/.env.kibana.token

# Password Generate

PW=$(head /dev/urandom | tr -dc A-Za-z0-9 | head -c 16

;)

ELASTIC\_PASSWORD="${ELASTIC\_PASSWORD:-$PW}" export ELASTIC\_PASSWORD



find $OUTPUT\_DIR -type f -exec chmod 655 -- {} + printf

"=====================================================\n

" printf "SSL Certifications generation completed successfully.\n" printf

"=====================================================\n "

# Setup-keystore.sh









# Create Keystore printf "========== Creating Elasticsearch Keystore



==========\n" printf "=====================================================\n

" elasticsearch-keystore create >>

/dev/null

# Setting Secrets and Bootstrap Password sh

/setup/keystore.sh echo "Elastic Bootstrap Password is:

$ELASTIC\_PASSWORD"

# Generating Kibana Token echo "Generating Kibana Service Token..."

# Delete old token if exists

/usr/share/elasticsearch/bin/elasticsearch-servicetokens delete elastic/kibana default &> /dev/null || true

# Generate new token TOKEN=$(/usr/share/elasticsearch/bin/elasticsearchservice- tokens create elastic/kibana default | cut -d

'=' -f2 | tr -d ' ') echo "Kibana Service Token is: $TOKEN" echo "KIBANA\_SERVICE\_ACCOUNT\_TOKEN=$TOKEN" >

$OUTPUT\_KIBANA\_TOKEN



# Replace current Keystore if [ -f "$OUTPUT\_KEYSTORE" ]; then





echo "Remove old elasticsearch.keystore"





rm $OUTPUT\_KEYSTORE fi echo "Saving new elasticsearch.keystore" mkdir -p "$(dirname $OUTPUT\_KEYSTORE)" mv $GENERATED\_KEYSTORE $OUTPUT\_KEYSTORE chmod 0644 $OUTPUT\_KEYSTORE



# Replace current Service Tokens File if [ -f "$OUTPUT\_SERVICE\_TOKENS" ]; then echo "Remove old service\_tokens file" rm

$OUTPUT\_SERVICE\_TOKENS fi echo "Saving new service\_tokens file" mv $GENERATED\_SERVICE\_TOKENS

$OUTPUT\_SERVICE\_TOKENS chmod 0644

$OUTPUT\_SERVICE\_TOKENS

printf "======= Keystore setup completed successfully

=======\n"

printf "=====================================================\n

" printf "Remember to restart the stack, or reload secure settings if changed settings are hot- reloadable.\n" printf "About Reloading Settings: https:/[/www.e](http://www.elastic.co/guide/en/elasticsearch/reference/)l[astic.co/guide/en/elasticsearch/reference/](http://www.elastic.co/guide/en/elasticsearch/reference/) current/secure-settings.html#reloadable- securesettings\n"



printf

"=====================================================\n

" printf "Your 'elastic' user password is:

$ELASTIC\_PASSWORD\n" printf "Your Kibana Service Token is: $TOKEN\n" printf

"=====================================================\n "

==============================X================================

