Deploying ELK Stack On Docker Container project source code

DONE BY : PRIYANKA DAS

docker-compose.logs.yml

```yaml
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

```yaml
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:
```

```yaml
services:
  elasticsearch-1:
    image: elastdocker/elasticsearch:${ELK_VERSION}
    build:
      context: elasticsearch/
      args:
        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:
```

```yaml
      - 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:
      context: elasticsearch/
      args:
        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}
```

```yaml
      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

```yaml
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

```yaml
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_VERSION}

# 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
```

```yaml
# each container and log type.
filebeat.autodiscover:
  providers:
    - type: docker
      templates:
        - condition:
            contains:
              docker.container.image: elasticsearch
          config:
            - 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
```

```yaml
                    paths:
'/var/lib/docker/containers/${data.docker.container.id}/
*.log'
              deprecation:
                input:
                  type: container
                  paths:
'/var/lib/docker/containers/${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:
```

```yaml
                  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:
```

```yaml
    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 {
        hosts => "${ELASTICSEARCH_HOST_PORT}"
        user => "${ELASTIC_USERNAME}"
        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
```

```bash
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
find $OUTPUT_DIR -type f -exec chmod 655 -- {} +

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

Setup-keystore.sh

```
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
```

```bash
# 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-service-
tokens delete elastic/kibana default &> /dev/null ||
true

# Generate new token
TOKEN=$(/usr/share/elasticsearch/bin/elasticsearch-
service-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
```

```bash
    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.elastic.co/guide/en/elasticsearch/reference/
current/secure-settings.html#reloadable-secure-
settings\n"
```

```
printf
"=======================================================\n
"
printf "Your 'elastic' user password is:
$ELASTIC_PASSWORD\n"
printf "Your Kibana Service Token is: $TOKEN\n"
printf
"=======================================================\n
"
```

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