#================================== Description ========================================

# Filebeat Config to send Elasticsearch/Logstash/Kibana in a docker host to Elasticsea-

# sh cluster.

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:

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

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:

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

---

## Default Kibana configuration from Kibana base image.

## https://github.com/elastic/kibana/blob/master/src/dev/build/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

---

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

# For per pipeline config, check docs: https://www.elastic.co/guide/en/logstash/current/logstash-settings-file.html

- pipeline.id: main

path.config: "/usr/share/logstash/pipeline/main.conf"

queue.type: memory

#- pipeline.id: second\_pipeline

# path.config: "/usr/share/logstash/pipeline/second.conf"

# queue.type: persisted

# pipeline.batch.size: 125

# queue.page\_capacity: 50mb

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"

}

}

ARG ELK\_VERSION

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

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

HEALTHCHECK --interval=240s --timeout=120s --retries=5 \

CMD curl -s -XGET 'http://127.0.0.1:9600'

# Add your logstash plugins setup here

# Example: RUN logstash-plugin install logstash-filter-json

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"

}

}

instances:

- name: elasticsearch

dns:

- elasticsearch

- localhost

ip:

- 127.0.0.1

- name: kibana

dns:

- kibana

- localhost

ip:

- 127.0.0.1

# Exit on Error

set -e

# Setting Bootstrap Password

echo "Setting bootstrap.password..."

(echo "$ELASTIC\_PASSWORD" | elasticsearch-keystore add -x 'bootstrap.password')

# ----- Setting Secrets

## Add Additional Config

# 1- Copy the below commented block, uncomment it, and replace <name>, <key>, and <KEY\_ENV\_VALUE>.

# 2- Pass <KEY\_ENV\_VALUE> to setup container in `docker-compose-setup.yml`

## Setting <name>

#echo "Setting <name>..."

#(echo "$<KEY\_ENV\_VALUE>" | elasticsearch-keystore add -x '<key>')

# ----- Setting S3 Secrets

## Setting S3 Access Key

#echo "Setting S3 Access Key..."

#(echo "$AWS\_ACCESS\_KEY\_ID" | elasticsearch-keystore add -x 's3.client.default.access\_key')

#

## Setting S3 Secret Key

#echo "Setting S3 Secret Key..."

#(echo "$AWS\_SECRET\_ACCESS\_KEY" | elasticsearch-keystore add -x 's3.client.default.secret\_key')

# Exit on Error

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

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

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

printf "SSL Certifications generation completed successfully.\n"

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

# Exit on Error

set -e

GENERATED\_KEYSTORE=/usr/share/elasticsearch/config/elasticsearch.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

# 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

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"