معرفی و نصب ابزار ELK و Filebeat

بهار ۱۳۹۸

فهرست

مقدمه

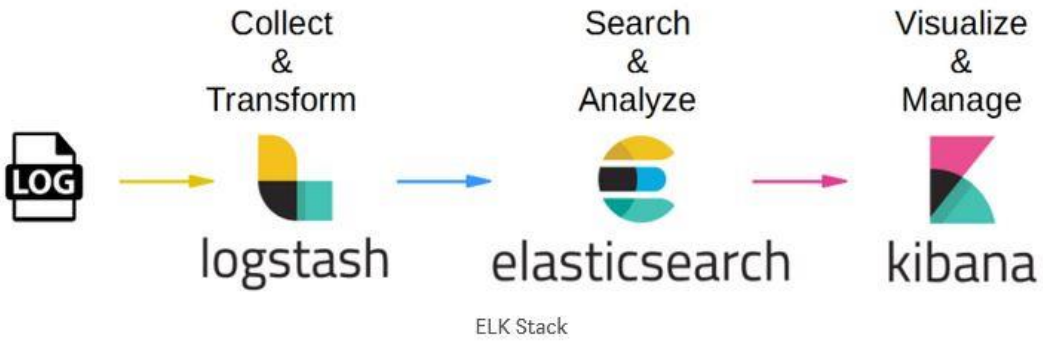
یکی از محبوب ترین و قدرتمند ترین ابزارهای مشاهده و تحلیل لاگ اپلیکیشن ها، ابزار ELK است که توسط شرکت Elastic ارائه شده است. این ابزار شامل سه برنامه اپن سورس Elasticsearch, Logstash و Kibana است.

برای لاگ گیری از کانتینرهای داکر می توان از دستور docker logs –f container-ID-or-Name استفاده کرد. اما لاگی که به ما تحویل داده می شود یک فایل بصورت متنی و خام است و نمی توان فیلتری برروی آن تعریف کرد. این دستور زمانی که تعداد کانتینرها کم باشد و به سرور داکر دسترسی وجود داشته باشد می تواند تا حدی جوابگو باشد. اما زمانی که تعداد کانتینرها زیاد باشد و بخواهیم بصورت متمرکز و یکپارچه لاگ کانتینرها را در جای دیگری خارج از سرور بررسی کنیم، این روش جوابگو نیست. یکی از راه حل هایی که برای این مساله وجود دارد این است که لاگ تمام کانتینرها را در جایی تجمیع نماییم و از طریق ابزاری گرافیکی به لاگها دسترسی پیدا کنیم. پکیج Elastic Stack یکی از محبوب ترین ابزارهای حل چنین مشکلاتی است که با نام ELK نیز شناخته می شود.

در اینجا ما از ELK برای نمایش لاگ کانتینرهای داکر استفاده می نماییم. و همچنین ابزارهای مورد نیاز را تماما به صورت ایمیج داکر نصب و تنظیم خواهیم نمود. بنابراین برروی سروری که می خواهیم ابزارها را را نصب نماییم از قبل باید docker و docker-compose نصب باشند. در اینجا ما ابزارها را برروی سرور CICD (192.168.253.75) نصب می کنیم (کانتینر اپلیکیشن ها نیز برروی همین سرور در حال اجرا است).

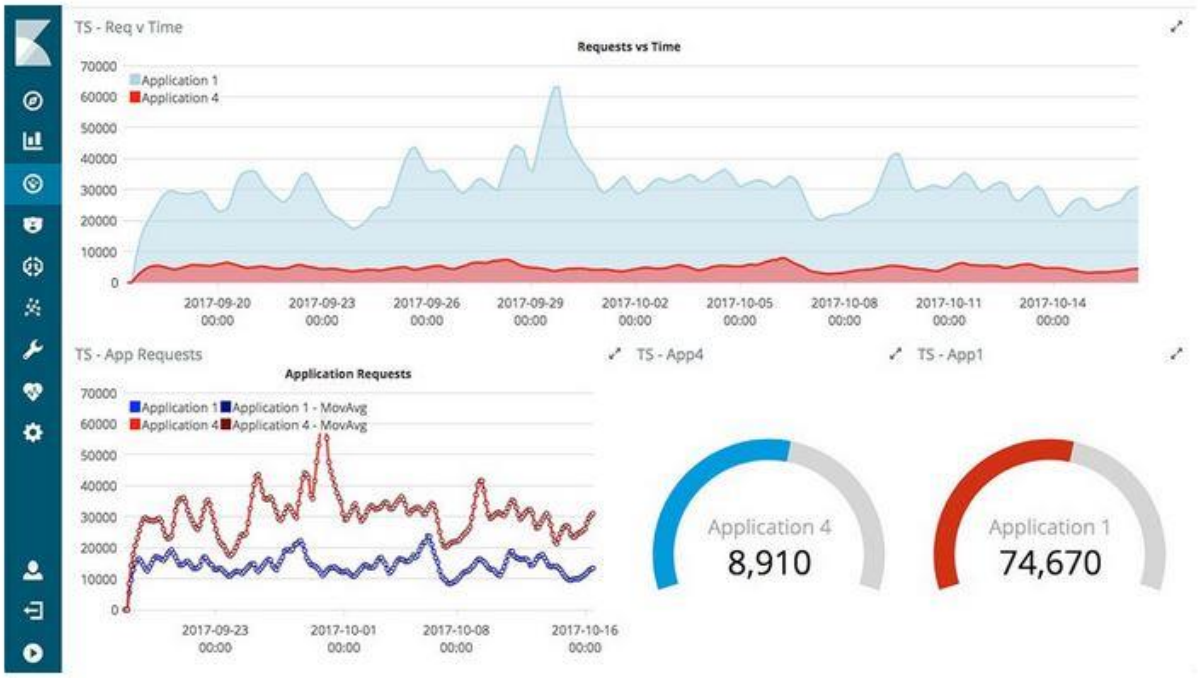
روش کار ELK:

در شکل ۱ طریقه کار ELK بصورت کلی نشان داده شده است.



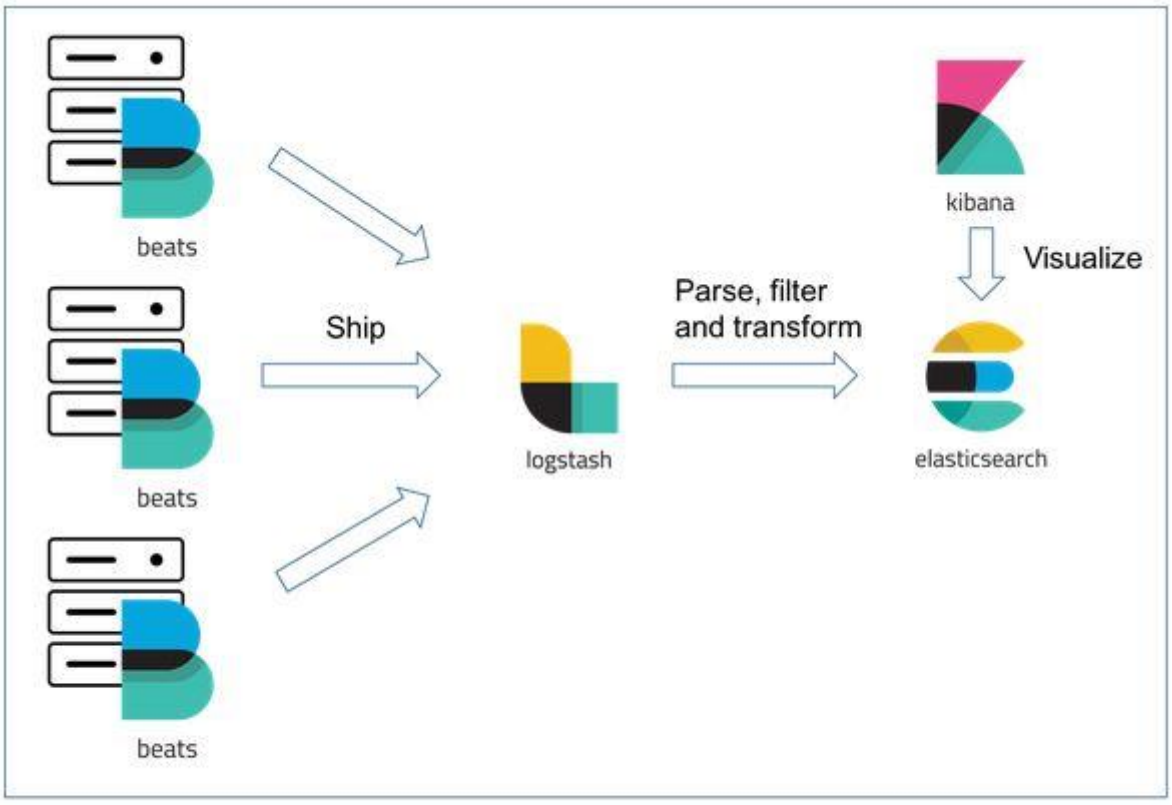
شکل ۱ نحوه کار ELK

همانگونه که در شکل مشخص است ابتدا لاگ ها به ابزار Logstash تحویل داده می شود: Logstash یک چرخه پردازش داده Server-side است که دیتا را بصورت همزمان از منابع مختلف دریافت نموده، آنها را Parse نموده و بعد از Transform به Elasticsearch ارسال می کند. Elasticsearch یک موتور توزیع شده جستجو و آنالیز است، که یک API قدرتمند RESTful JSON-based را فراهم می کند. این ابزار قلب پشته ELK است که به صورت متمرکز لاگ ها را ذخیره می کند. در نهایت Kibana لاگ ها را از Elasticsearch گرفته و بصورت گرافیکی آن ها را نمایش می دهد. ابزار Kibana یک ابزار گرافیکی است که شامل امکانات متنوعی برای فیلتر کردن، جستجو و تحلیل لاگ ها است. در شکل ۲ نتیجه تحلیلی در Kibana نشان داده شده است.



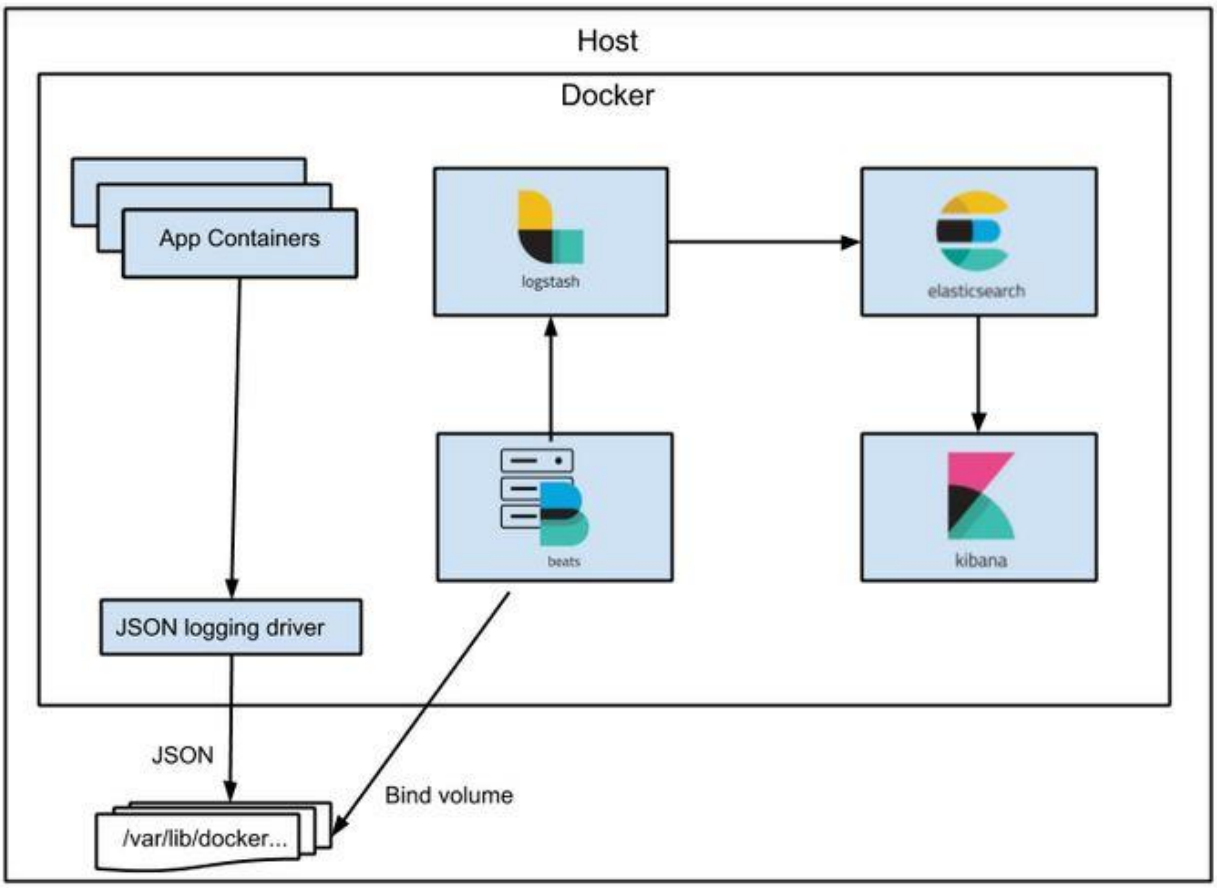
شکل ۲ امکانات تحلیل لاگ در Kibana

بصورت پیش فرض لاگ کانتینرهای داکر در دایرکتوری /var/lib/docker/containers/{container-id}/{container-id}-json.log و با فرمت json ذخیره می شود. بنابراین برای استفاده از ابزار ELK تنها کافیست به گونه ای لاگ ها از دایرکتوری مربوط به هر کانتینر گرفته شود و در اختیار Logstash قرار گیرد. ابزاری که می توان برای این کار از آن استفاده کرد، ابزار Filebeat است این ابزار یک shipper برای جمع آوری لاگ ها بوده و یکی دیگر از محصولات شرکت Elastic است. بنابراین روال جمع آوری و نمایش گرافیکی لاگ اپلیکیشن های داکری مطابق شکل ۳ خواهد بود.



شکل ۳ ارتباط بین ابزارهای مورد نیاز برای تحلیل لاگ

در نتیجه برای جمع آوری و تحلیل لاگ های اپلیکیشن های داکر آدرس لاگ های داکر را بصورت /var/lib/docker/containers/*/*-json.log در تنظیمات Filebeat قرار می دهیم. همچنین Metadata مربوط به کانتینرها را نیز می توان از /var/run/docker.sock دریافت کرد. بنابراین آنچه که قرار است انجام دهیم مطابق شکل ۴ خواهد بود.



شکل ۴ مراحل دسترسی و نمایش لاگ اپلیکیشن ها

نصب ابزارها و تنظیمات

تنظیمات مربوط به ابزارها در پروژه /bh-framework/baharan-framework-devops قرار داده شده است.

برای اجرای ابزارهای ELK و Filebeat ابتدا آخرین نسخه ایمیج هرکدام را از سایت Elastic یا از Docker-hub ، Pull می نماییم. در شکل ۵ ایمیج های این ابزار ها برروی سرور CICD را نشان می دهد.

```
192.168.253.10:8082/logstash          7.1.1     b0cb1543380d   6 weeks ago   847MB
192.168.253.10:8082/kibana            7.1.1     67f17df6ca3e   6 weeks ago   746MB
elk-ok-filebeat_kibana                latest    67f17df6ca3e   6 weeks ago   746MB
192.168.253.10:8082/elasticsearch     7.1.1     b0e9f9f047e6   6 weeks ago   894MB
elk-ok-filebeat_elasticsearch         latest    b0e9f9f047e6   6 weeks ago   894MB
docker.elastic.co/beats/filebeat      7.1.1     0bd69a03e199   6 weeks ago   288MB
```

شکل ۵ ایمیج مربوط به ابزارهای تحلیل لاگ

سپس باید برای هرکدام از این ابزارها یک فایل config.yml و همچنین dockerFile (در هنگام اجرای docker-compose از روی ایمیج هایی که دانلود کرده ایم ، ایمیج دلخواه می سازیم) و در نهایت فایل docker-compose.yml را تهیه می کنیم تا ارتباط بین ابزارها و ویژگی ها و محدودیت های هرکدام را تعیین نماییم.

تنظیمات Elasticsearch

فایل Dockerfile مربوط به elasticsearch را مطابق زیر تهیه می کنیم:

```
FROM 192.168.253.10:8082/elasticsearch:7.1.1
```

فایل تنظیمات آن را نیز به اسم elasticsearch.yml مطابق زیر آماده کرده و آن را در دایرکتوری config قرار می دهیم:

```
cluster.name: "docker-cluster"
network.host: 0.0.0.0
discovery.zen.minimum_master_nodes: 1
discovery.type: single-node
```

تنظیمات Logstash

فایل Dockerfile مطابق زیر:

```
FROM 192.168.253.10:8082/logstash:7.1.1
```

و فایل logstash.yml در دایرکتوری config مطابق زیر:

```
http.host: "0.0.0.0"
path.config: /usr/share/logstash/pipeline
```

همچنین فایل logstash.conf در دایرکتوری pipeline مطابق زیر:

```
input {
        beats {
          port => 5044
          host => "0.0.0.0"
```

```
        }
}
output {
        elasticsearch {
                hosts => "elasticsearch:9200"
        }
}
```

تنظیمات Kibana

فایل Dockerfile شامل

```
FROM 192.168.253.10:8082/kibana:7.1.1
```

فایل kibana.yml در دایرکتوری config و شامل:

```
server.name: kibana
server.host: "0"
elasticsearch.hosts: ["http://elasticsearch:9200"]
```

تنظیمات filebeat

فایل Dockerfile شامل

```
FROM docker.elastic.co/beats/filebeat:7.1.1

# Copy our custom configuration file
COPY filebeat.yml /usr/share/filebeat/filebeat.yml

USER root
# Create a directory to map volume with all docker log files
RUN mkdir /usr/share/filebeat/dockerlogs
RUN chown -R root /usr/share/filebeat/
RUN chmod -R go-w /usr/share/filebeat/
```

فایل filebeat.yml شامل:

```
filebeat.inputs:
- type: docker
  combine_partial: true
  containers:
```

```
    path: "/usr/share/dockerlogs/data"
    stream: "stdout"
    ids:
      - "*"
  exclude_files: ['\.gz$']
  ignore_older: 10m

processors:
  # decode the log field (sub JSON document) if JSON encoded, then maps it's fields to elasticsearch fields
- decode_json_fields:
    fields: ["log", "message"]
    target: ""
    # overwrite existing target elasticsearch fields while decoding json fields
    overwrite_keys: true
- add_docker_metadata:
    host: "unix:///var/run/docker.sock"

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

# setup filebeat to send output to logstash
output.logstash:
  hosts: ["logstash:5044"]

# Write Filebeat own logs only to file to avoid catching them with itself in docker log files
logging.level: error
logging.to_files: false
logging.to_syslog: false
loggins.metrice.enabled: false
logging.files:
  path: /var/log/filebeat
  name: filebeat
  keepfiles: 7
  permissions: 0644
ssl.verification_mode: none
```

تنظیم فایل docker-compose

در نهایت باید فایل docker-compose.yml را برای اجرای ابزارها تنظیم نماییم. برای این کار یک فایل docker-compose.yml ایجاد کرده و مقادیر زیر را درون آن قرار می دهیم:

```
version: '2.2'
services:
```

```
  elasticsearch:
    build:
      context: elasticsearch/
    volumes:
      -
./elasticsearch/config/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml:ro
    ports:
      - "9200:9200"
      - "9300:9300"
    environment:
      ES_JAVA_OPTS: "-Xmx256m -Xms256m"
    networks:
      - elk
    cpus: 0.5
    mem_limit: 2G
  logstash:
    build:
      context: logstash/
    volumes:
      - ./logstash/config/logstash.yml:/usr/share/logstash/config/logstash.yml:ro
      - ./logstash/pipeline:/usr/share/logstash/pipeline:ro
    ports:
      - "5044:5044"
    environment:
      LS_JAVA_OPTS: "-Xmx256m -Xms256m"
    networks:
      - elk
    depends_on:
      - elasticsearch
    cpus: 0.5
    mem_limit: 2G
   # cpu_count: 1

  kibana:
    build:
      context: kibana/
    volumes:
      - ./kibana/config/:/usr/share/kibana/config:ro
    ports:
      - "5601:5601"
    networks:
      - elk
    depends_on:
      - elasticsearch
   # cpu_count: 1
    cpus: 0.5
```

```
    mem_limit: 2G

  filebeat:
    build:
      context: filebeat/
    volumes:
      # needed to access all docker logs (read only) :
     - "/var/lib/docker/containers:/usr/share/dockerlogs/data:ro"
      # needed to access additional informations about containers
     - "/var/run/docker.sock:/var/run/docker.sock"
    networks:
      - elk
    #depends_on:
     # - elasticsearch
    links:
       - logstash
    #cpu_count: 1
    cpus: 0.5
    mem_limit: 2G
networks:
  elk:
    driver: bridge
```

در نهایت تنها کافیست به دایرکتوری فایل docker-compose.yml رفته و با اجرای docker-compose up -d ابزارها اجرا خواهند شد. همانطور که در فایل docker-compose مشخص کرده ایم. کنسول kibana برروی پورت 5601 اجرا خواهد شد. بنابراین کافیست در مرورگر آدرس سرور CICD را با پورت 5601 وارد نماییم تا به کنسول kibana دسترسی پیدا کنیم. (192.168.253.75:5601). در مستندهای دیگر نحوه کار با Kibana و چگونگی ذخیره لاگ از آن توضیح داده شده است.

فایل هایی که توضیح داده شد را می توانید از پروژه bh-framework/baharan-framework-devops دریافت نمایید.