

**DOCKER COMPOSE LAB**

1. **Overview**

* Through this lab, everyone can understanding:
  + How to deploy docker, docker compose
  + How to write docker compose yaml file

1. **Prepare**

* 01 server Linux: CentOS 7.0
* Editor tool: Visual Studio Code
* Ready connectivity
* Terminal tool: MobarXterm
* FTP tool: WinSCP

1. **How to implement**
   1. **Install docker**
   2. **Install docker compose**$ 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  
        
      $ sudo chmod +x /usr/local/bin/docker-compose  
      $ docker-compose --version
   3. **Make yaml file**

* Open Visual Studio Code application, and write/make docker-compose file as content below
* After, save with file name: ***docker-compose.yaml***

version: '3.7'

services:

  elasticsearch:

    build:

      context: elasticsearch/

      args:

        ELK\_VERSION: $ELK\_VERSION

    volumes:

      - type: bind

        source: ./elasticsearch/config/elasticsearch.yml

        target: /usr/share/elasticsearch/config/elasticsearch.yml

        read\_only: true

      - type: volume

        source: elasticsearch

        target: /usr/share/elasticsearch/data

    ports:

      - "9200:9200"

      - "9300:9300"

    environment:

      ES\_JAVA\_OPTS: "-Xmx256m -Xms256m"

      ELASTIC\_PASSWORD: 123456a@

      # Use single node discovery in order to disable production mode and avoid bootstrap checks

      # see https://www.elastic.co/guide/en/elasticsearch/reference/current/bootstrap-checks.html

      discovery.type: single-node

    networks:

      - elk

  logstash:

    build:

      context: logstash/

      args:

        ELK\_VERSION: $ELK\_VERSION

    volumes:

      - type: bind

        source: ./logstash/config/logstash.yml

        target: /usr/share/logstash/config/logstash.yml

        read\_only: true

      - type: bind

        source: ./logstash/pipeline

        target: /usr/share/logstash/pipeline

        read\_only: true

    ports:

      - "5000:5000/tcp"

      - "5000:5000/udp"

      - "9600:9600"

    environment:

      LS\_JAVA\_OPTS: "-Xmx256m -Xms256m"

    networks:

      - elk

    depends\_on:

      - elasticsearch

  kibana:

    build:

      context: kibana/

      args:

        ELK\_VERSION: $ELK\_VERSION

    volumes:

      - type: bind

        source: ./kibana/config/kibana.yml

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

        read\_only: true

    ports:

      - "5601:5601"

    networks:

      - elk

    depends\_on:

      - elasticsearch

networks:

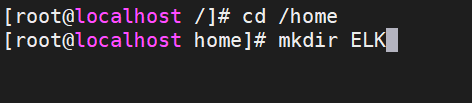
  elk:

    driver: bridge

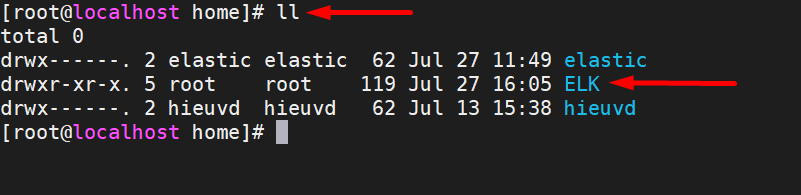
volumes:

  elasticsearch:

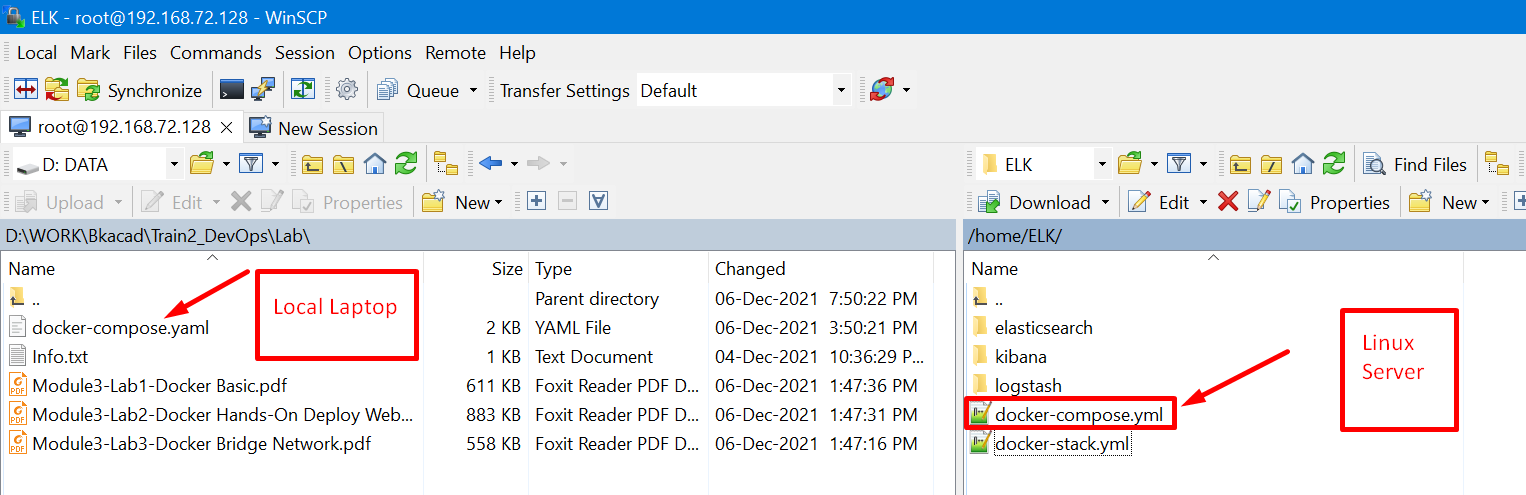
* 1. **Create working directory**
* On Linux server, create working directory at /home, using command as:



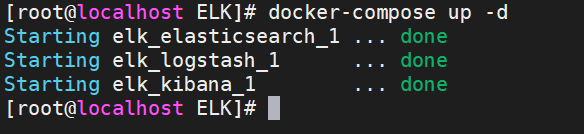
* Verify working directory



* Using WinSCP, transfer ***docker-compose.yaml*** file to ***ELK*** directory

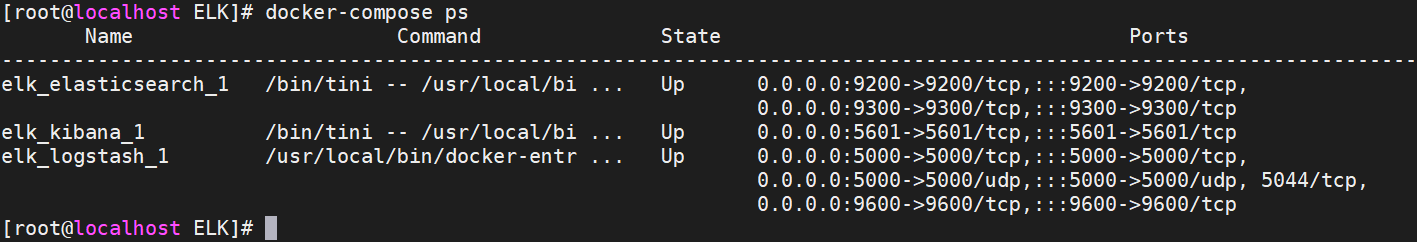


* 1. **Deployment**
* At working directory ELK, start this command: *docker-compose up -d*

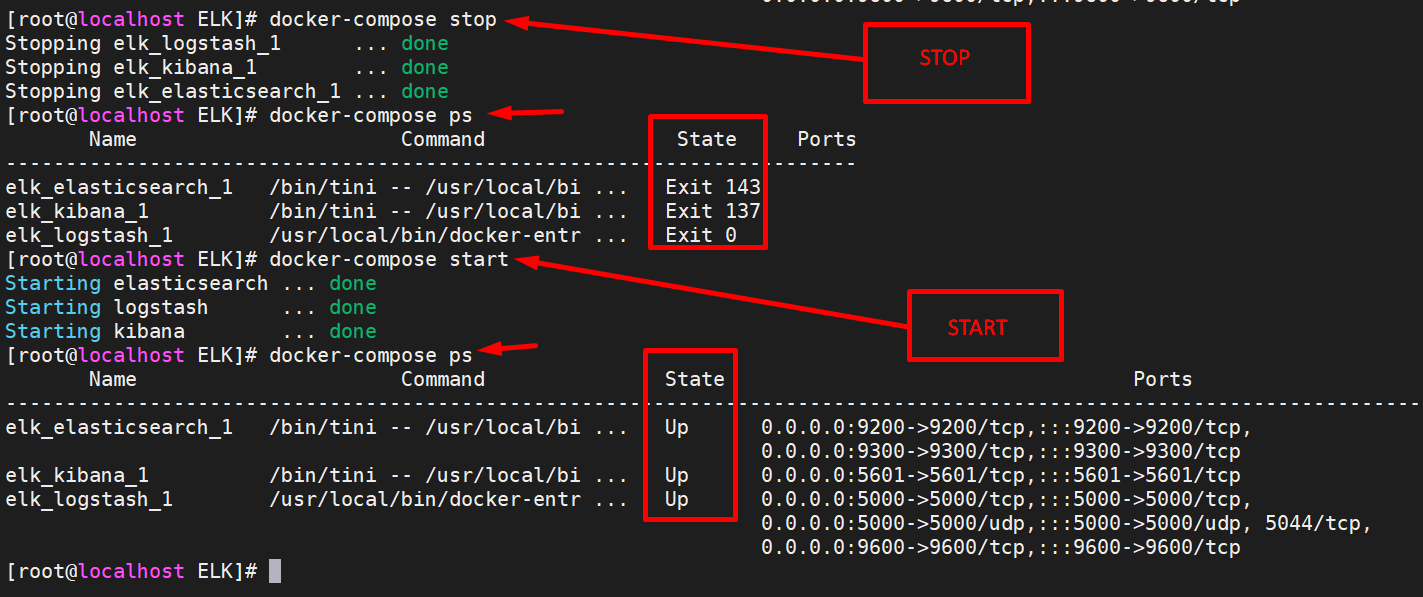


1. **Verify**

* Check and confirm status of docker containers: *docker-compose ps*

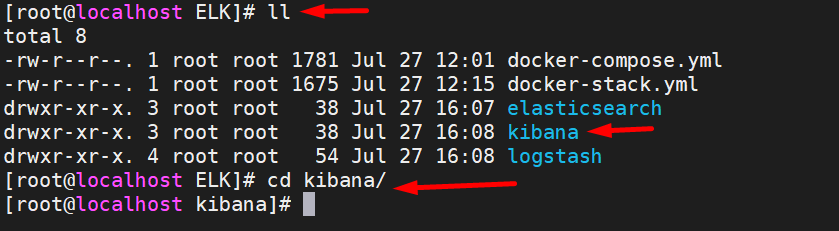


* In addition, to stop/start docker containers, perform by these command below

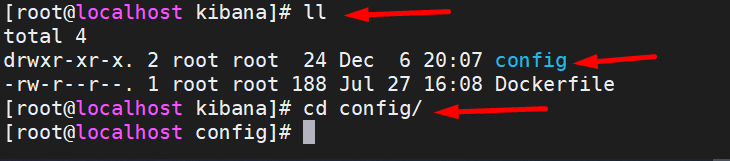


1. **Configuration**
   1. **Kibana config**

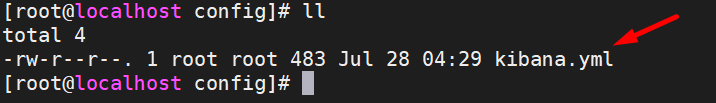
* Access to ***kibana*** directory

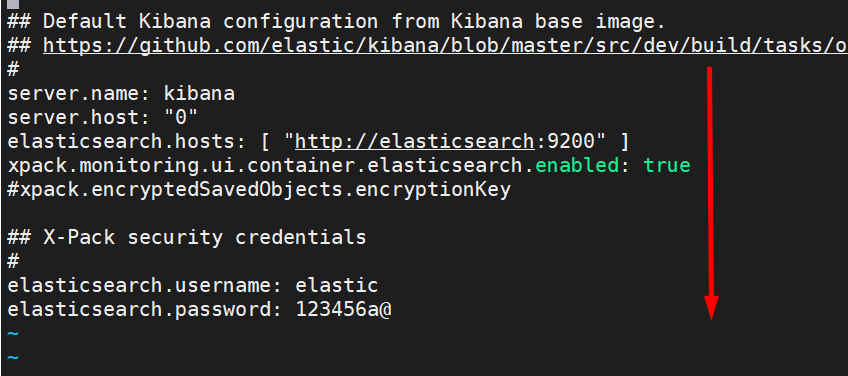


* Access to ***config*** directory (into ***kibana***)

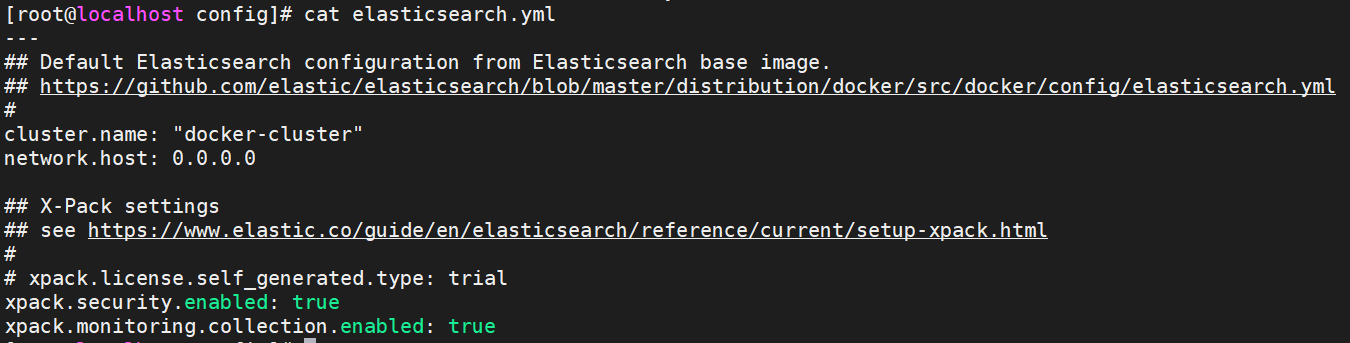


* Update configuration for this file ***kibana.yaml***, after save this file

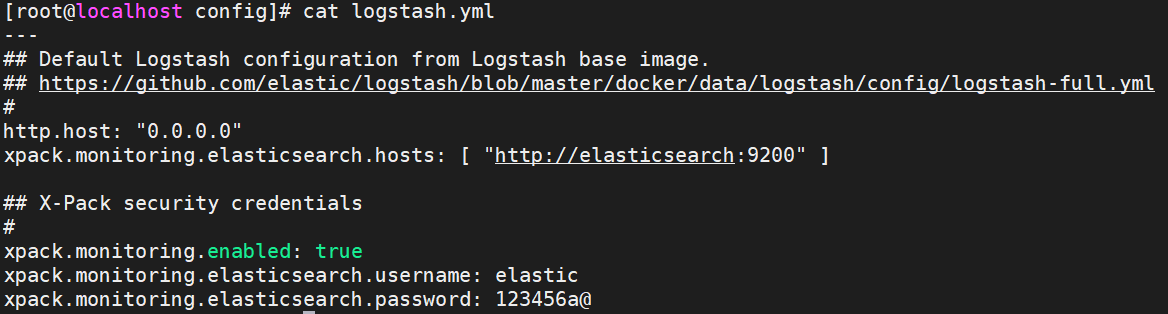




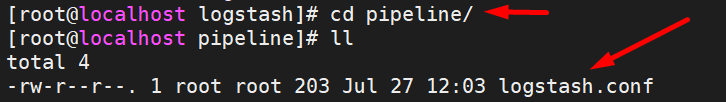
* 1. **Elastic config**
* Do the same as Kibana and update configuration for file ***elasticsearch.yml***



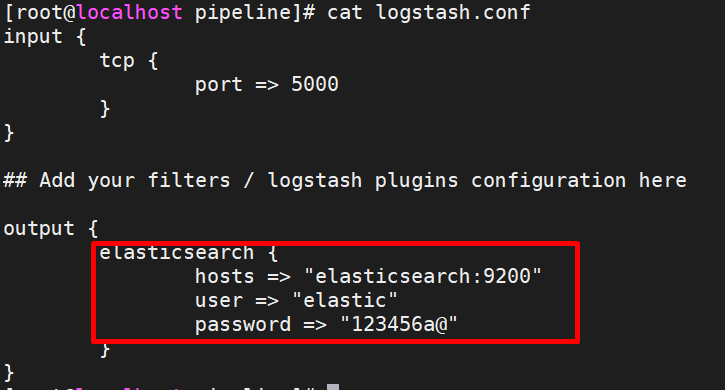
* 1. **Logstash config**
* Do the same as Kibana and update configuration for file ***logstash.yml***



* Access to ***pipeline*** *directory*

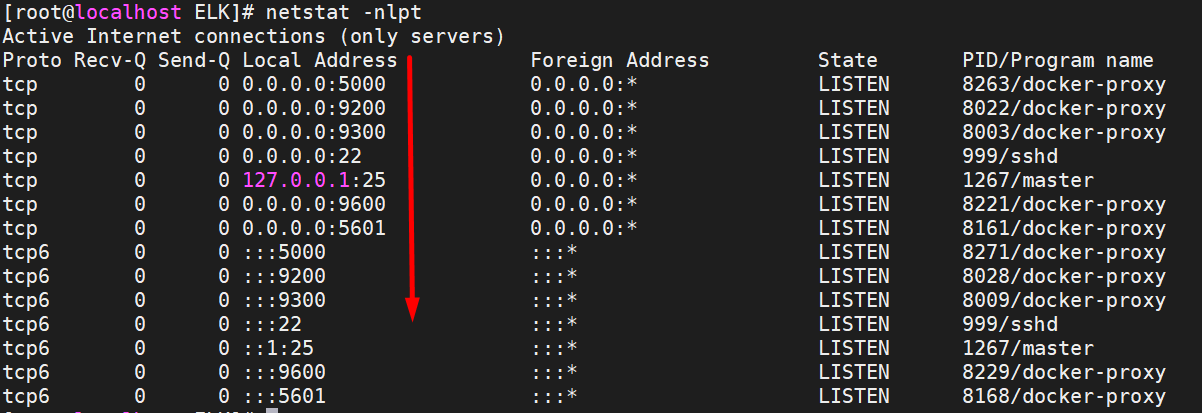


* Confirm and update value for file ***logstash.conf***



1. **Confirm ELK stack**

* Confirm again about port listening



* Browse access to Kibaba interface by IP address of docker host

