Sesión 4 : swarm (ELK+TIG)

resumen

```
docker ps
docker ps -f name=ep_devops_mysql.1
docker ps -f name=ep_devops_mysql.1 -q
docker exec -ti $(docker ps -f name=ep_devops_mysql.1 -q ) /bin/bash
docker logs $(docker ps -f name=ep_devops_mysql.1 -q )
docker logs -f $(docker ps -f name=ep_devops_mysql.1 -q )
```

/via: https://lpic-devops.pad.floss.cat/p/docker-troubleshooting

TIG (Telegraf+Influxdb+Grafana)

- crear base de datos en influxdb: `curl -i -XPOST http://localhost:8086/query –data-urlencode «q=CREATE DATABASE telegraf`
- borrar base de datos: `curl -i -XPOST http://localhost:8086/query –data-urlencode «q=DROP DATABASE telegraf`
- retencion_policy a XX dias para que purge los días y no se limite a X entradas
- importar dashboards: https://grafana.com/dashboards

Grafana

- dashboards:
  - visión cluster
  - visión stack y servicios
  - contenedores
- paneles: exportar JSON
- plugin: Diagram (pinta un servicio y estados según valores de métrica)

edición panel

- los elementos que se meten en 1 row se pueden repetir, en función de las variables
- ```
  $host = show tag values with key = "host"
  $stack =     show tag values with key = "com.docker.stack.namespace"
  $service = show tag values with key = "com.docker.swarm.service.name" WHERE "com.docker.stack.namespace" =~ /^$stack/
  $container = show tag values with key = "container_name" WHERE "host" =~ /^$host$/ AND "com.docker.stack.namespace" =~ /^$stack/  AND "com.docker.swarm.service.name" =~ /^$service/
  ```

prometheus

- = Grafana + Alertas (aunque Grafana ahora ya si)
- node exporter : telegraf
  ```

- prometheus: base datos + alertas
  - https://stefanprodan.com/2016/a-monitoring-solution-for-docker-hosts-containers-and-containerized-services/

ELK

swarm-ELK.yml

```
#       Docker Stack to deploy ELK + Logspout
#       Based on .......
#       Updated by: Kenneth Peiruza, kenneth@floss.cat
#       Sun Mar  4 13:15:47 CET 2018
#
#      cluster.name: 'docker-cluster'
#      bootstrap.memory_lock: 'true'
version: '3.4'

services:
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch-oss:6.2.2
    environment:
      ES_JAVA_OPTS: '-Xms768m -Xmx768m'
    networks:
      - elasticsearch
    volumes:
      -
${REMOTE_BIND}/${STACK_NAME}/elasticsearch/data:/usr/share/elasticsearch/data
    deploy:
      replicas: 1

  logstash:
    image: docker.elastic.co/logstash/logstash-oss:6.2.2
    volumes:
      -
${REMOTE_BIND}/${STACK_NAME}/logstash/config:/usr/share/logstash/pipeline/
    depends_on:
      - elasticsearch
    networks:
      - elasticsearch
      - logstash
    deploy:
      replicas: 1

  logspout:
    image: bekt/logspout-logstash
    environment:
      ROUTE_URIS: 'logstash://logstash:5000'
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    depends_on:
      - logstash
    networks:
      - logstash
    deploy:
```

```
      mode: global
      restart_policy:
        condition: on-failure
        delay: 30s

  kibana:
    image: docker.elastic.co/kibana/kibana-oss:6.2.2
    ports:
      - 5601:5601
    depends_on:
      - elasticsearch
    networks:
      - elasticsearch
      - proxy
    environment:
      ELASTICSEARCH_URL: 'http://elasticsearch:9200'
    deploy:
      replicas: 1
      labels:
        traefik.port: 5601
        traefik.frontend.rule: "Host:${LOGS_URL}"
        traefik.docker.network: "proxy"

networks:
  default:
    driver: 'overlay'
  logstash:
    driver: 'overlay'
  elasticsearch:
    driver: 'overlay'
  proxy:
    external: true
```

```
input {
  udp {
    port  => 5000
    codec => json
  }
}

filter {
  if [docker][image] =~ /logstash/ {
    drop { }
  }
}

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

+ info:

- Filtros Logstash: https://www.elastic.co/guide/en/logstash/current/config-examples.html
- Filtros grok: https://logz.io/blog/logstash-grok/