

Sesión 3

varios

- spread brain?:
 - 3 nodos develop
 - 5 nodos producción
- `sudo hostnamectl set-hostname <nombre_máquina>`
- docker mysql: importar los fichero SQL de la carpeta **/docker-entrypoint-initdb.d/** automaticamente
 - visto en el Dockerfile
(<https://github.com/docker-library/mysql/blob/a7a737f1eb44db467c85c8229df9d886dd63460e/8.0/Dockerfile>)
- directiva **constrains** para deployar en máquinas concretas

swarm

- `docker swarm init`
 - para crear un swarm con varios nodos, solo 1 ha de iniciarlo
- `docker swarm join-token {manager|worker}`
 - podemos añadirnos como manager o como workers al swarm creado
- `docker stack deploy -c stack-portainer.yml portainer`
- `docker stack ls : stacks`
- `docker stack services <stack> : nombre servicios del stack`
- `docker service ps <servicio> :`
- `docker node inspect`
- `ssh <nodo_swarm>`
 - `docker ps -f name=<nombre_contenedor>`
- `docker service logs <servicio>`
- si no puedes ver logs del contenedor, mirar en logs de docker:
 - `/var/log/messages`
 - `/var/log/syslog`
 - 0...

portainer

[swarm-portainer.yml](#)

```
version: '3.4'

services:
  portainer:
    image: portainer/portainer
    ports:
      - 9000:9000
    networks:
      - proxy
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - /srv/docker/portainer/data:/data
    deploy:
```

```
mode: replicated
replicas: 1
labels:
  traefik.port: 9000
  traefik.frontend.rule: 'Host:portainer.midominio.com'
  traefik.docker.network: 'proxy'
placement:
  constraints: [node.role == manager]
networks:
  proxy:
    external: true
```

wordpress

swarm-wp+mysql.yml

```
version: '3.3'
services:
  db:
    image: mysql:5.7
    volumes:
      - /ruta/local/bbdd:/var/lib/mysql
    environment:
      MYSQL_ROOT_PASSWORD: somewordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress

  wordpress:
    image: wordpress:latest
    ports:
      - "8000:80"
    volumes:
      - /ruta/local/web:/var/www/html
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
```

influx+grafana+telegraf

stack-triolalala.yml

```
version: '3.4'

services:
  influxdb:
    image: influxdb:latest
    command:
```

```
-config /etc/influxdb/influxdb.conf
environment:
  INFLUXDB_ADMIN_USER: admin
  INFLUXDB_ADMIN_PASSWORD: admin123
networks:
  - agents
  - grafana
volumes:
  - /srv/docker/influxdb/data:/var/lib/influxdb
  - /srv/docker/influxdb/config:/etc/influxdb/config:ro
deploy:
  replicas: 1

telegraf:
  image: telegraf:latest
  environment:
    HOST_PROC: '/rootfs/proc'
    HOST_SYS: '/rootfs/sys'
    HOST_ETC: '/rootfs/etc'
  networks:
    - agents
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock:ro
    - /sys:/rootfs/sys:ro
    - /proc:/rootfs/proc:ro
    - /run:/rootfs/run:ro
    - /etc:/rootfs/etc:ro
    - /srv/docker/influxdb/config:/etc/telegraf/
  deploy:
    mode: global # Metes 1 instancia en cada nodo
    restart_policy:
      condition: on-failure
      delay: 5s

grafana:
  image: grafana/grafana
  environment:
    GF_INSTALL_PLUGINS: 'grafana-clock-panel,grafana-piechart-panel,grafana-simple-json-datasource'
  volumes:
    - /srv/docker/influxdb/grafana/data:/var/lib/grafana/
  networks:
    - grafana
    - proxy
  depends_on:
    - influxdb
  deploy:
    replicas: 1
  ports:
    - 9002:3000

networks:
  agents:
    external: true
  grafana:
    external: true
  proxy:
```

```
external: true
```

stack-kpeiruza.yml

```
# Author: Kenneth Peiruza
version: '3.4'

services:
  influxdb:
    image: library/influxdb:latest
    command: -config /etc/influxdb/influxdb.conf
    environment:
      INFLUXDB_ADMIN_USER: ${INFLUXDB_ADMIN_USER:-admin}
      INFLUXDB_ADMIN_PASSWORD: ${INFLUXDB_ADMIN_PASSWORD:-admin}
    networks:
      - agents
      - grafana
    volumes:
      - ${REMOTE_MOUNT}/${STACK_NAME}/influxdb/data:/var/lib/influxdb
      - ${REMOTE_MOUNT}/${STACK_NAME}/influxdb/config:/etc/influxdb/config:ro
    deploy:
      replicas: 1

  grafana:
    image: grafana/grafana
    environment:
      GF_INSTALL_PLUGINS: 'grafana-clock-panel,grafana-piechart-panel,grafana-simple-json-datasource'
    volumes:
      - ${REMOTE_MOUNT}/${STACK_NAME}/grafana/data:/var/lib/grafana/
    networks:
      - grafana
      - proxy
    depends_on:
      - influxdb
    deploy:
      replicas: 1
    labels:
      traefik.port: 3000
      traefik.frontend.rule: "Host:${TRAEFIK_FQDN}"
      traefik.docker.network: "proxy"

# Fancy replacing placement on a static node. Add Net-storage and drop this out

  telegraf:
    image: telegraf/telegraf
    environment:
      HOST_PROC: '/rootfs/proc'
      HOST_SYS: '/rootfs/sys'
      HOST_ETC: '/rootfs/etc'
    networks:
      - agents
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
```

```

- /sys:/rootfs/sys:ro
- /proc:/rootfs/proc:ro
- /run:/rootfs/run:ro
- /etc:/rootfs/etc:ro
- /etc/telegraf/:/etc/telegraf/
deploy:
  mode: global # Metes 1 instancia en cada nodo
  restart_policy:
    condition: on-failure
    delay: 5s

networks:
  agents:
    driver: 'overlay'
  grafana:
    driver: 'overlay'
  proxy:
    external: true

```

- **telegraf.conf:**

- escupe fichero de config base: `docker run -ti library/telegraf telegraf config`
- modificado el **inputs.docker**:
 - descomentado endpoint, gather_services
- y el **outputs.influxdb**
 - descomentado urls,database, skip_database_creation

influxdb.conf

```

reporting-disabled = false
bind-address = "127.0.0.1:8088"

[meta]
  dir = "/var/lib/influxdb/meta"
  retention-autocreate = true
  logging-enabled = true

[data]
  dir = "/var/lib/influxdb/data"
  index-version = "inmem"
  wal-dir = "/var/lib/influxdb/wal"
  wal-fsync-delay = "0s"
  query-log-enabled = true
  cache-max-memory-size = 1073741824
  cache-snapshot-memory-size = 26214400
  cache-snapshot-write-cold-duration = "10m0s"
  compact-full-write-cold-duration = "4h0m0s"
  max-series-per-database = 1000000
  max-values-per-tag = 100000
  max-concurrent-compactions = 0
  trace-logging-enabled = false

[coordinator]
  write-timeout = "10s"
  max-concurrent-queries = 0
  query-timeout = "0s"

```

```
log-queries-after = "0s"
max-select-point = 0
max-select-series = 0
max-select-buckets = 0

[retention]
enabled = true
check-interval = "30m0s"

[shard-precreation]
enabled = true
check-interval = "10m0s"
advance-period = "30m0s"

[monitor]
store-enabled = true
store-database = "_internal"
store-interval = "10s"

[subscriber]
enabled = true
http-timeout = "30s"
insecure-skip-verify = false
ca-certs = ""
write-concurrency = 40
write-buffer-size = 1000

[http]
enabled = true
bind-address = ":8086"
auth-enabled = false
log-enabled = true
write-tracing = false
pprof-enabled = true
https-enabled = false
https-certificate = "/etc/ssl/influxdb.pem"
https-private-key = ""
max-row-limit = 0
max-connection-limit = 0
shared-secret = ""
realm = "InfluxDB"
unix-socket-enabled = false
bind-socket = "/var/run/influxdb.sock"

[[graphite]]
enabled = false
bind-address = ":2003"
database = "graphite"
retention-policy = ""
protocol = "tcp"
batch-size = 5000
batch-pending = 10
batch-timeout = "1s"
consistency-level = "one"
separator = "."
udp-read-buffer = 0
```

```
[[collectd]]
  enabled = false
  bind-address = ":25826"
  database = "collectd"
  retention-policy = ""
  batch-size = 5000
  batch-pending = 10
  batch-timeout = "10s"
  read-buffer = 0
  typesdb = "/usr/share/collectd/types.db"
  security-level = "none"
  auth-file = "/etc/collectd/auth_file"

[[opentsdb]]
  enabled = false
  bind-address = ":4242"
  database = "opentsdb"
  retention-policy = ""
  consistency-level = "one"
  tls-enabled = false
  certificate = "/etc/ssl/influxdb.pem"
  batch-size = 1000
  batch-pending = 5
  batch-timeout = "1s"
  log-point-errors = true

[[udp]]
  enabled = false
  bind-address = ":8089"
  database = "udp"
  retention-policy = ""
  batch-size = 5000
  batch-pending = 10
  read-buffer = 0
  batch-timeout = "1s"
  precision = ""

[continuous_queries]
  log-enabled = true
  enabled = true
  run-interval = "1s"
```

relevante:

- constrains
- tasks.<servicio>
- si apuntas un loadbalancer (traefik) a los master...
- en el caso de este portainer, no desplegamos en al red de docker, si no sobre la máquina loca (pila tcp-ip local)

```
version: '3.4'
```

```
services:
```

```
  portainer:
```

```
    image: portainer/portainer
```

```
command: '-H "tcp://tasks.agent:9001" --tlsskipverify'
ports:
  - ${PUBLIC_PORT}:9000
networks:
  - proxy
  - portainer_agent
volumes:
  - /var/run/docker.sock:/var/run/docker.sock
  - ${REMOTE_MOUNT}/${STACK_NAME}/portainer/data:/data
deploy:
  mode: replicated
  replicas: 1
  labels:
    traefik.port: 9000
    traefik.frontend.rule: "Host:${FQDN}"
    traefik.docker.network: 'proxy'
  placement:
    constraints: [node.role == manager]
```

```
agent:
  image: portainer/agent
  environment:
    AGENT_CLUSTER_ADDR: tasks.agent
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock
  ports:
    - target: 9001
      published: 9001
      protocol: tcp
      mode: host
  networks:
    - portainer_agent
  deploy:
    mode: global
    placement:
      constraints: [node.platform.os == linux]
```

```
networks:
  portainer_agent:
    external: true
  proxy:
    external: true
```

traefik (con https y let's encrypt)

[traefik-proxy.yaml](#)

```
version: '3.5'

services:
  traefik:
    image: library/traefik:1.7
```

```

command:
  - "--api"
  - "--entrypoints=Name:http Address::80 Redirect.EntryPoint:https"
  - "--entrypoints=Name:https Address::443 TLS"
  - "--defaultentrypoints=http,https"
  - "--acme"
  - "--acme.entryPoint=https"
  - "--acme.httpChallenge.entryPoint=http"
  - "--acme.OnHostRule=true"
  - "--acme.onDemand=false"
  - "--acme.email=info@example.com"
  - "--acme.storage=/etc/traefik/acme/acme.json"
  - "--docker"
  - "--docker.swarmmode"
  - "--docker.domain=example.com"
  - "--docker.watch"
# - "--logLevel=DEBUG"
# - "--traefikLog.filePath=/dev/sterr"
# - "--accessLog.filePath=/dev/stdout"
# - "--accessLog.filters.statusCodes=200,300-302,400-460,500"
# - "--accessLog.filters.retryAttempts=true"
# - "--accessLog.fields.defaultMode=keep"
# - "--accessLog.fields.names=Username=drop"
# - "--accessLog.fields.headers.defaultMode=keep"
# - "--accessLog.fields.headers.names=User-Agent=redact
Authorization=drop Content-Type=keep"
# - "--traefikLog.format=json"
# - "--accessLog.format=json"
networks:
  - frontal
volumes:
  - /var/run/docker.sock:/var/run/docker.sock
  - /srv/docker/traefik/certificates:/etc/traefik/acme/
ports:
  - 80:80
  - 443:443
  - 8080:8080
deploy:
  restart_policy:
    condition: on-failure

networks:
  frontal:
    external: true

```

traefik (simple)

```

version: '3.5'

services:
  traefik:
    image: library/traefik:1.7
    command:
      - "--api"

```

```
- "--entrypoints=Name:http Address::80"
- "--defaultentrypoints=http"
- "--docker"
- "--docker.swarmmode"
- "--docker.domain=example.com"
- "--docker.watch"
# - "--logLevel=DEBUG"
# - "--traefikLog.filePath=/dev/sterr"
# - "--accessLog.filePath=/dev/stdout"
# - "--accessLog.filters.statusCodes=200,300-302,400-460,500"
# - "--accessLog.filters.retryAttempts=true"
# - "--accessLog.fields.defaultMode=keep"
# - "--accessLog.fields.names=Username=drop"
# - "--accessLog.fields.headers.defaultMode=keep"
# - "--accessLog.fields.headers.names=User-Agent=redact Authorization=drop
Content-Type=keep"
# - "--traefikLog.format=json"
# - "--accessLog.format=json"
networks:
  - frontal
volumes:
  - /var/run/docker.sock:/var/run/docker.sock
  - /srv/docker/traefik/certificates:/etc/traefik/acme/
ports:
  - 80:80
  - 443:443
  - 8080:8080
deploy:
  restart_policy:
    condition: on-failure

networks:
  frontal:
    external: true
```

From: <https://miguelangel.torresegea.es/wiki/> - miguel angel torres egea

Permanent link: <https://miguelangel.torresegea.es/wiki/info: cursos: pue: devops: sesion3?rev=1550305476>

Last update: 16/02/2019 00:24

