Experiment 10

Roll no: R171218009

AIM: Working with Metadata, Log File using Docker

10 A) Metadata & Labels

Labels can be attached to containers when they are launched via `docker run`

assign a label called user with an ID to the container. This would allow us to query for all the containers running related to that particular user.

```
$ docker run -l user=12345 -d redis
9f2bb67433420c70b8f6e22b5a0c90a419716c0e3bde3e028025eaf948a40f31
$
```

If you're adding multiple labels, then this can be done using an external file.

```
$ echo 'user=123461' >> labels && echo 'role=cache' >> labels
$
```

The *--label-file=<filename>* option will create a label for each line in the file.

```
$ docker run --label-file=labels -d redis
5b9a4fcf402fd02baebaffde2e97b83bee476ac92b6f58234c1f45488f2f4c0f
$
```

Inspecting a container named ‘rd’

```
$ docker run --label-file=labels -d redis
5b9a4fcf402fd02baebaffde2e97b83bee476ac92b6f58234c1f45488f2f4c0f
$ docker inspect rd
[
    {
        "Id": "4782fd9036c87e4cfea3f6ae4634e5fa3f6db23758366ea6f396cbc05d38b2a4",
        "Created": "2021-05-09T14:37:17.830760243Z",
        "Path": "docker-entrypoint.sh",
        "Args": [
            "redis-server"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 968,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2021-05-09T14:37:18.501115562Z",
            "FinishedAt": "0001-01-01T00:00:00Z"
        },
        "Image": "sha256:4e8db158f18dc71307f95260e532df39a9b604b51d4e697468e82845c50cfe28",
        "ResolvConfPath": "/var/lib/docker/containers/4782fd9036c87e4cfea3f6ae4634e5fa3f6db23758
38b2a4/resolv.conf",
        "HostnamePath": "/var/lib/docker/containers/4782fd9036c87e4cfea3f6ae4634e5fa3f6db2375836
```

filter the JSON response to just the Labels section

```
$ docker inspect -f "{{json .Config.Labels }}" rd
{"com.katacoda.created":"automatically","com.katacoda.private-msg":"magic","user":"scrapbook"}
$
```

Query by label

```
$ docker ps --filter "label=user=scrapbook"
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
            NAMES
4782fd9036c8        redis               "docker-entrypoint.s…"   9 minutes ago       Up 9 minutes        6379/tc
p           rd
```

10 B) Log Files

there is an instance of Redis running with the name *redis-server*.

```
$ docker run -d --name redis-server redis
f0ce921172e530bb3bb3bd8824647644b0833ccb275211d5f06e9f5e4fcfbb24
$
$
```

Using the Docker client, we can access the standard out and standard error outputs using docker logs

```
$ docker logs redis-server
1:C 09 May 14:50:18.438 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
1:C 09 May 14:50:18.439 # Redis version=4.0.11, bits=64, commit=00000000, modified=0, pid=1, just started
1:C 09 May 14:50:18.439 # Warning: no config file specified, using the default config. In order to specify a co
fig file use redis-server /path/to/redis.conf
1:M 09 May 14:50:18.442 * Running mode=standalone, port=6379.
1:M 09 May 14:50:18.442 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/
somaxconn is set to the lower value of 128.
1:M 09 May 14:50:18.442 # Server initialized
1:M 09 May 14:50:18.442 # WARNING overcommit_memory is set to 0! Background save may fail under low memory cond
tion. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run the command '
ysctl vm.overcommit_memory=1' for this to take effect.
1:M 09 May 14:50:18.442 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This wi
l create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kerne
/mm/transparent_hugepage/enabled' as root, and add it to your /etc/rc.local in order to retain the setting afte
 a reboot. Redis must be restarted after THP is disabled.
1:M 09 May 14:50:18.442 * Ready to accept connections
$
```

Syslog: syslog is a widely used standard for message logging. It permits separation of the software that generates messages, the system that stores them, and the software that reports and analyses them.

redirecting the redis logs to syslog:

```
$ docker run -d --name redis-syslog --log-driver=syslog redis
f29004760f4724f63ed64adc3249cd1f49d3a9f0ae6c8c903241b56c11981107
```

These logs can be accessed via the syslog stream.

Disable logging:

This is particularly useful for containers which are very verbose in their logging.

When the container is launched simply set the log-driver to none.

```
$ docker run -d --name redis-none --log-driver=none redis
de4000457bcaf8aa7f60df32c5afdba29501f800cd04d5dffa9b6aea0f224e8c
$
```

Using inspect command, output only LogConfig section for all three containers we created

```
$ docker inspect --format '{{ .HostConfig.LogConfig }}' redis-server
{json-file map[]}
```

```
$ docker inspect --format '{{ .HostConfig.LogConfig }}' redis-syslog
{syslog map[]}
```

```
$ docker inspect --format '{{ .HostConfig.LogConfig }}' redis-none
{none map[]}
```