AARHUS UNIVERSITET

Microservices and DevOps

Scalable Microservices

Docker Health Check

Henrik Bærbak Christensen

Nygard

- *Health Check:*
  - A health check is a webpage/API call that reveals the application's internal view of its own health. [Nygard, p 169]

    - IP address, version of runtime, service version, status (accepting work), state (connection pools, caches, circuit breakers, …)

- That is,
  - Instead of (in addition to) the centralized monitoring system reporting health of a service, the service *itself* provides it
    - Not really of any use, in case the service has failed…

AARHUS UNIVERSITET

Ex: /health of CaveService

- Example: Added a health path to my cave service
  - Pretty easy, just another route with relatively hard-coded data…

```
^Ccsdev@m1:~/proj/cave$ http localhost:9999/msdo/v1/cave/health
HTTP/1.1 200 OK
Content-Type: application/json
Date: Thu, 04 Nov 2021 08:19:00 GMT
Server: Jetty(9.4.31.v20200723)
Transfer-Encoding: chunked

{
    "state": "healthy",
    "title": "CaveService by Henrik Bærbak",
    "upSince": "2021-11-04T09:19:00.855618+01:00[Europe/Copenhagen]",
    "version": "v1"
}
```

AARHUS UNIVERSITET

Docker Support

- Your Dockerfile supports container healthchecks
  - Ex: Check every 5 min that main page is responding in < 3secs

```
HEALTHCHECK --interval=5m --timeout=3s \
  CMD curl -f http://localhost/ || exit 1
```

  - This is a rudimentary check, better to provide a /health path with e.g. JSON payload that describe health – like my previous example

- Of course, your application must then provide that particular /health path !

AARHUS UNIVERSITET

Docker Support

When a container has a healthcheck specified, it has a *health status* in addition to its normal status. This status is initially `starting`. Whenever a health check passes, it becomes `healthy` (whatever state it was previously in). After a certain number of consecutive failures, it becomes `unhealthy`.

The options that can appear before `CMD` are:

- `--interval=DURATION` (default: `30s`)
- `--timeout=DURATION` (default: `30s`)
- `--start-period=DURATION` (default: `0s`)
- `--retries=N` (default: `3`)

The health check will first run **interval** seconds after the container is started, and then again **interval** seconds after each previous check completes.

- If it does not appear ‘healthy’ after the N retries, the container stops

AARHUS UNIVERSITET

Use in Swarm

- The swarm will *restart* a container (given section ‘restart-policy’) if it is not healthy
- Compose-file may overrule the healthcheck (or add it)

```
healthcheck:
  test: ["CMD", "curl", "-f", "http://localhost"]
  interval: 1m30s
  timeout: 10s
  retries: 3
  start_period: 40s
```

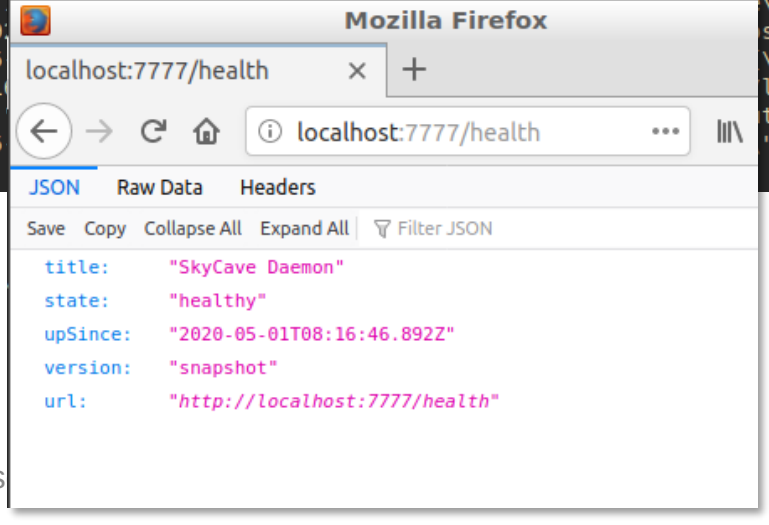
Example

AARHUS UNIVERSITET

/health on Daemon

```
csdev@m51:~/proj/cave$ docker run --name cave -d -p 7777:7777 henrikbaerbak/private:cave
258ede4265e46ac2caa4fc978c5b19610707d83929355cda874d6f1da4516162
csdev@m51:~/proj/cave$ docker inspect --format='{{json .State.Health}}' cave
{"Status":"starting","FailingStreak":0,"Log":[]}
csdev@m51:~/proj/cave$ docker inspect --format='{{json .State.Health}}' cave
{"Status":"starting","FailingStreak":0,"Log":[]}
csdev@m51:~/proj/cave$ docker inspect --format='{{json .State.Health}}' cave
{"Status":"starting","FailingStreak":0,"Log":[{"Start":"2020-05-01T10:16:15.522517449+02:00","End":"2020-05-01T10:16:15.879777926+02:00","ExitCode":1,"Output":""}]}
csdev@m51:~/proj/cave$ docker inspect --format='{{json .State.Health}}' cave
{"Status":"starting","FailingStreak":0,"Log":[{"Start":"2020-05-01T10:16:15.522517449+02:00","End":"2020-05-01T10:16:15.879777926+02:00","ExitCode":1,"Output":""}]}
```

```
csdev@m51:~/proj/cave$ docker inspect --format='{{json .State.Health}}' cave
{"Status":"healthy","FailingStreak":0,"Log":[{"Start":"2020-05-01T10:16:15.522517449+02:00","End":"2020-05-01T10:16:15.879777926+02:00","ExitCode":1,"Output":""},{"Start":"2020-05-01T10:16:45.882349199+02:00","End":"2020-05-01T10:16:46.230605475+02:00","ExitCode":1,"Output":""},{"Start":"2020-05-01T10:17:16.244181463+02:00","End":"2020-05-01T10:
":\"SkyCave Daemon\",\"state\":\"healthy\",\"upSince\":\"2020-05-01T08:16:46.89
t:7777/health\"}"},{"Start":"2020-05-01T10:17:46.54392539+02:00","End":"2020-05
"title\":\"SkyCave Daemon\",\"state\":\"healthy\",\"upSince\":\"2020-05-01T08:1
ocalhost:7777/health\"}"},{"Start":"2020-05-01T10:18:16.762468762+02:00","End":
put":"{\"title\":\"SkyCave Daemon\",\"state\":\"healthy\",\"upSince\":\"2020-05
http://localhost:7777/health\"}"}]}
```



Healthcheck in Dockerfile

AARHUS UNIVERSITET

Health in Compose

- Added healthcheck in compose on my CaveService

```
csdev@m51:~/proj/cave$ docker ps -a
CONTAINER ID        IMAGE                              COMMAND                  CREATED             STATUS                    PORTS
98396882859a        henrikbaerbak/private:cave         "./gradlew daemon21 …"   4 minutes ago       Up 4 minutes              7777/tcp
yt2dq36v9l3ozg08ows
76d5f7d94051        henrikbaerbak/private:caveservice  "java -jar /root/cav…"   4 minutes ago       Up 4 minutes (healthy)    9999/tcp
```

- Added path '/halt' that stops the web server!

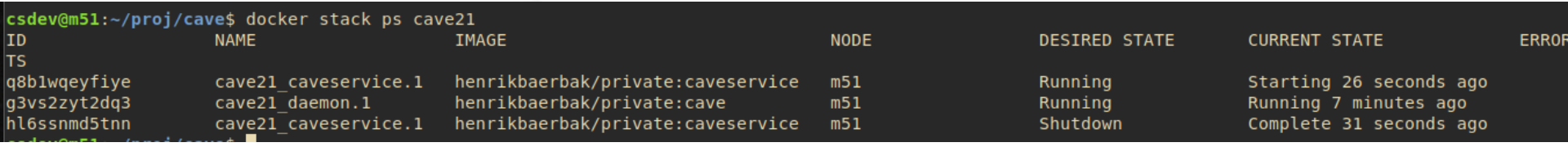
```
csdev@m51:~/proj/cave$ http localhost:9999/halt
HTTP/1.1 200 OK
Content-Type: text/html;charset=utf-8
Date: Fri, 01 May 2020 11:02:49 GMT
Server: Jetty(9.4.z-SNAPSHOT)
Transfer-Encoding: chunked

<h1>HALTING<h1>
```

AARHUS UNIVERSITET

Health in Compose

```
deploy:
  replicas: 1
  restart_policy:
    delay: 5s
    max_attempts: 15
```



```
csdev@m51:~/proj/cave$ docker stack ps cave21
ID                  NAME                  IMAGE                                NODE                DESIRED STATE       CURRENT STATE            ERROR
TS
q8b1wqeyfiye        cave21_caveservice.1  henrikbaerbak/private:caveservice    m51                 Running             Starting 26 seconds ago
g3vs2zyt2dq3        cave21_daemon.1       henrikbaerbak/private:cave           m51                 Running             Running 7 minutes ago
hl6ssnmd5tnn        cave21_caveservice.1  henrikbaerbak/private:caveservice    m51                 Shutdown            Complete 31 seconds ago
```

- Morale: Simple ‘control plane’ behavior is possible in swarm…

AARHUS UNIVERSITET

A failure experiment

- I tried in my ‘strangled’ cave consisting of
  - Daemon + CaveService
- To view the startup process..
- It seems that…
  - (though I find the docs pretty ambigous)
  - The container is not ‘ready for action’ until the first healthcheck has passed…
  - Which means my Daemon did not work until the interval period had expired…

```
# Do health checks and restart if down
healthcheck:
  test: ["CMD", "curl", "-f", "http://localhost:9999/msdo/v1/cave/health"]
  interval: 30s
  timeout: 5s
  retries: 3
  start_period: 10s
```

AARHUS UNIVERSITET

Summary

- Healthchecks can serve two purposes in a container context
  - As a web page we can inspect to review ‘interesting stuff’
    - Ala…
  - As a container mechanism to control restarts and simple monitoring
  - Caveat: ‘curl’ is itself a pretty big package to require to be installed in the container…