Szolgáltatások skálázása - reverse proxy segítségével

https://github.com/knehez/isi - docker_example_2

A **HAProxy** egy nyílt forráskódú, magas rendelkezésre állású TCP/HTTP terheléselosztó és proxy szerver. A HAProxy segítségével egyetlen IP-cím mögötti több backend szerver között osztható el a terhelés, így javítva az alkalmazások teljesítményét és megbízhatóságát.

docker-compose.yml

```
version: "3.3"
services:
  web:
    build: .
    ports:
      - "5000"
  redis:
    image: "redis:alpine"
  haproxy:
    image: "haproxytech/haproxy-alpine:2.4"
    volumes:
        - ./haproxy.cfg:/usr/local/etc/haproxy/haproxy.cfg:ro
    depends_on:
        - web
    ports:
      - "80:80"
```

HAproxy config:

A következő konfiguráció beállítja a HAProxy-t arra, hogy a 80-as porton figyelje a bejövő HTTP forgalmat, és az öt backend szerver között osztsa szét, amelyek egészségi állapotát ellenőrzik, mielőtt a forgalmat továbbítanák. Továbbá beállít egy statisztikai felületet a 8404-es porton a HAProxy példány monitorozásához.

```
global
  stats socket /var/run/api.sock user haproxy group haproxy mode 660 level
admin expose-fd listeners
  log stdout format raw local0 info

defaults
  mode http
  timeout client 10s
  timeout connect 5s
  timeout server 10s
  timeout http-request 10s
  log global

frontend stats
  bind *:8404
  stats enable
```

```
  stats uri /
  stats refresh 10s

frontend myfrontend
  bind 0.0.0.0:80
  mode http
  default_backend webservers

backend webservers
  server s1 web:5000 check
  server s2 web:5000 check
  server s3 web:5000 check
  server s4 web:5000 check
  server s5 web:5000 check
```

1. A “global” szekcióban vannak olyan beállítások, amelyek a teljes HAProxy példányra vonatkoznak. Például megadja a statisztikai felület eléréséhez szükséges elérési útvonalat és jogosultságokat. Az stdout-on keresztül megjeleníti a naplóüzeneteket.
2. A “defaults” szekcióban a timeout értékek beállításával határozza meg, hogy mennyi időt szán a HAProxy a kapcsolatok felépítésére, a kliens kérések és a válaszok feldolgozására.
3. A “frontend stats” rész beállítja a statisztikai felületet, amelyen keresztül a felhasználók monitorozhatják a HAProxy állapotát. Megadja a figyelt portot, engedélyezi a statisztikai adatok megjelenítését és frissítési időt határoz meg.
4. A “frontend myfrontend” rész a HAProxy frontend részét határozza meg, ami azt jelenti, hogy ezen a ponton érkeznek a kérések a HAProxy szerverre. A példában a 80-as portra van beállítva a figyelés, és az összes kérés átirányításra kerül a “webservers” backendre.
5. A “backend webservers” részben a tényleges backend szerverek konfigurálását végzi, amelyek a tényleges kiszolgálást végzik. Itt az “s1”, “s2”, “s3”, “s4” és “s5” szerverek beállítása megtörténik, amelyek mindegyike a web:5000 címen futó szervereket ellenőrizni fogják a forgalom továbbítása előtt.

Indítás:

```
  docker-compose up --scale web=4
```

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:informacios_rendszerek_integralasa:docker_loadbalancer

Last update: **2025/03/25 16:01**