

# LAPORAN PRAKTIKUM SISTEM PARAREL TERDISTRIBUSI

## TOPIK : LOAD BALANCE PADA DOCKER DENGAN HAPROXY

Oleh: Umar Alf Syahran 23080960066

### 1. Struktur Folder.

```
~ home
├── sistem-pararel
│   ├── compose.yml
│   ├── haproxy.cfg
│   └── nginx
│       ├── site
│       └── index.html
└── caddy
    ├── site
    └── index.html
```

```
umar@uuu:~$ ls -R sistem-pararel/
sistem-pararel/:
caddy  compose.yml  haproxy.cfg  nginx  site

sistem-pararel/caddy:
site

sistem-pararel/caddy/site:
index.html

sistem-pararel/nginx:
site

sistem-pararel/nginx/site:
index.html
```

2. Pertama diawali dengan membuat direktori tempat konfigurasi docker compose dan file local tempat project yang akan disimpan.

```
umar@uuu:~$ mkdir sistem-pararel
umar@uuu:~$ cd sistem-pararel/
umar@uuu:~/sistem-pararel$ nano compose.yml
```

### 3. Mengisi file compose.yml.

```
umar@uuu:~/sistem-pararel$ cat compose.yml
version: "3.9"

services:

  haproxy:
    image: haproxy:latest
    container_name: lb
    ports:
      - "80:80"
      - "8404:8404"
    volumes:
      - ./haproxy.cfg:/usr/local/etc/haproxy/haproxy.cfg
    networks:
      - webnet

  caddy:
    image: caddy:latest
    container_name: caddy
    volumes:
      - ./caddy/site:/usr/share/caddy
    networks:
      - webnet

  nginx:
    image: nginx:latest
    container_name: nginx
    volumes:
      - ./nginx/site:/usr/share/nginx/html
    networks:
      - webnet

networks:
  webnet:
    driver: bridge
umar@uuu:~/sistem-pararel$
```

Menggunakan HAProxy untuk load balance, dan web server 1 menggunakan caddy sementara web server 2 menggunakan nginx di keduanya sudah diberi index.html untuk membedakan tampilan antar web server nantinya.

Untuk network menggunakan driver bridge karena cocok untuk load balance antar kontainer bisa saling ping tetapi host dari luar tidak bisa kecuali port nya di publish memberikan keamanan. Dan setiap kontainer akan mendapatkan IP masing masing sehingga masing masing web server terisolasi dimana load balance membutuhkan IP berbeda dan isolasi. Port 8404 digunakan untuk mengakses tabel stats milik HaProxy.

4. Dilanjutkan dengan membuat file konfigurasi load balance pada HAProxy di file haproxy.cfg.

```
umar@uuu:~/sistem-pararel$ cat haproxy.cfg
frontend ft_web
    bind *:80
    mode http
    default_backend web_servers
    # Jika user tidak mengirim data apa pun selama 30 detik
    # → HAProxy menutup koneksi.
    timeout client 30s

backend web_servers
    mode http
    balance roundrobin
    server caddy caddy:80 check
    server nginx nginx:80 check

    # Jika backend lebih dari 30 detik tidak beri reply → timeout.
    timeout server 30s

    # Jika dalam 5 detik tidak bisa connect (server down, network error)
    # → dianggap gagal lalu HAProxy coba server berikutnya.
    timeout connect 5s

listen stats
    bind *:8404
    mode http
    stats enable
    stats uri /stats
    stats refresh 5s

    timeout server 30s
    timeout connect 5s
    timeout client 30s
umar@uuu:~/sistem-pararel$
```

Untuk mengarahkan traffic, disini tidak perlu menuliskan IP. Seperti pada bagian server caddy dan server nginx, di situ menggunakan nama DNS caddy dan nginx. Dimana ini adalah nama kontainer.

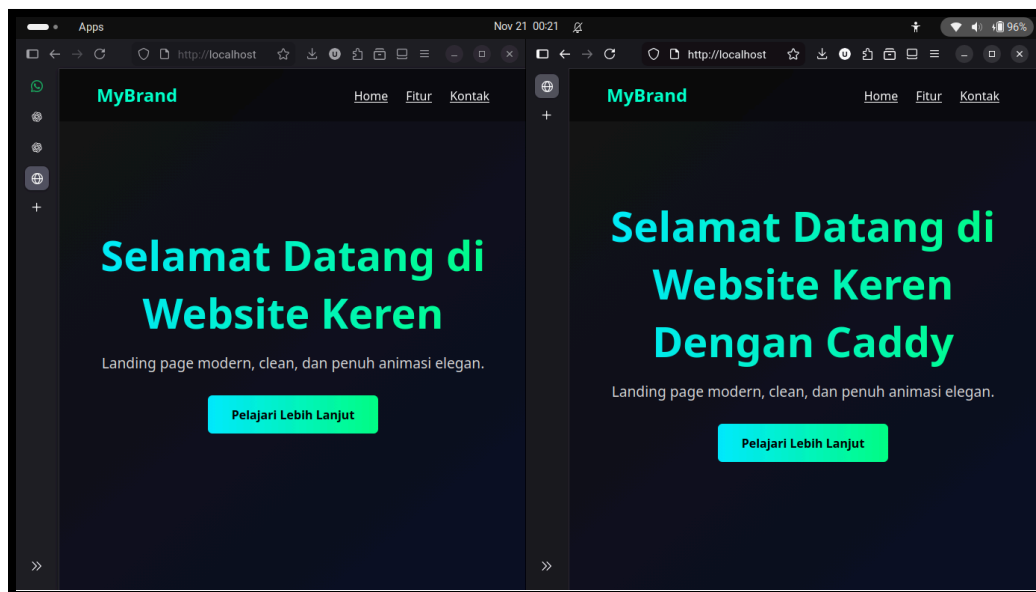
Itu semua berkat pengaturan networks dengan menggunakan driver = bridge. Di Haproxy harus membuat block timeout pada tiap bagian (frontend, backend, dan stats) + mode http. Jika tidak maka stats tidak akan bisa diakses.

5. Menjalankan docker compose untuk memulai semua kontainer yang sudah dibuat tadi.

```
umar@uuu:~/sistem-pararel$ docker compose up -d
WARN[0000] /home/umar/sistem-pararel/compose.yml:
tential confusion
[+] Running 4/4
✓ Network sistem-pararel_webnet Created
✓ Container lb Started
✓ Container caddy Started
✓ Container nginx Started
```

Output menunjukkan jika 3 kontainer berhasil berjalan dan 1 networks berhasil dibuat.

6. Mengetes load balance di host.



[HAProxy version 3.2.8-9200f39, released 2025/11/07](#)  
[Statistics Report for pid 10](#)

[General process information](#)

pid = 10 (process #1, nbproc = 1, nbthread = 4)  
 uptime = 0d 0h19m13s; warnings = 0  
 system limits: memmax = unlimited, dlimin = 524287  
 maxsock = 524287, maxconn = 262125, reached = 0, maxpipes = 0  
 current conns = 2; current pipes = 0/0; conn rate = 0/sec; bit rate = 211.003 kbps  
 Running tasks: 0/24 (0 nice); idle = 100 %

active UP  
 active UP, going down  
 active DOWN, going up  
 active or backup DOWN  
 active or backup UP  
 backup UP  
 backup UP, going down  
 backup DOWN, going up  
 not checked  
 active or backup DOWN for maintenance (MAINT)  
 active or backup SOFT STOPPED for maintenance  
 Note: "NOLE/7DRAIN" = UP with load-balancing disabled.

Display option:
 

- Scope:
- Hide DOWN servers
- Disable refresh
- Refresh interval
- CSV export
- JSON export (schema)

External resources:
 

- Primary site
- Secondary site
- Online manual

<div>front</div>																														
Queue			Session rate			Sessions			Bytes			Denied			Errors			Warnings			Status			Server						
Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LibTot	Last	In	Out	Req	Resp	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Down	Downtime	Thrtle		
Frontend																														
0	0	0	1	0	0	1	0	0	262125	1	1	3112	38632	0	0	0	0	0	0	OPEN										
<div>web servers</div>																														
Queue			Session rate			Sessions			Bytes			Denied			Errors			Warnings			Status			Server						
Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LibTot	Last	In	Out	Req	Resp	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Down	Downtime	Thrtle		
caddy																														
0	0	0	0	2	0	1	0	1	4	4	18m42s	1552	19400	0	0	0	0	0	0	19m13s UP	L4OK in 0ms	1/1	Y	-	0	0	0s	-		
nginx																														
0	0	0	0	2	0	1	0	1	4	4	18m41s	1560	19232	0	0	0	0	0	0	19m13s UP	L4OK in 0ms	1/1	Y	-	0	0	0s	-		
Backend																														
0	0	0	0	3	0	1	0	1	26213	8	8	18m41s	3112	38632	0	0	0	0	0	0	19m13s UP		2/2	2	0	0	0s			
<div>stats</div>																														
Queue			Session rate			Sessions			Bytes			Denied			Errors			Warnings			Status			Server						
Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LibTot	Last	In	Out	Req	Resp	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Down	Downtime	Thrtle		
Frontend																														
0	0	0	1	0	0	2	2	2	262125	3	3	186406	10970988	0	0	0	0	0	0	OPEN										
Backend																														
0	0	0	0	0	0	0	0	0	26213	0	0	186406	10970988	0	0	0	0	0	0	19m13s UP		0/0	0	0	0	0	0s			

Saya mencoba refresh dan langsung berubah tampilannya sesuai dengan index.html yang saya buat tiap web server. Kesimpulannya saya sudah berhasil melaksanakan praktik load balance menggunakan HAProxy dengan webserver Caddy + Nginx. Yeayyyy !!