

```

1 Lab9. Container Network
2
3 1. Container Network 사용하기
4 1)docker0 사용 확인하기
5 $ ip addr
6 1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
7 link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
8 inet 127.0.0.1/8 scope host lo
9 valid_lft forever preferred_lft forever
10 inet6 ::1/128 scope host
11 valid_lft forever preferred_lft forever
12 2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen
13 1000
14 link/ether 02:6b:81:64:e1:32 brd ff:ff:ff:ff:ff:ff
15 inet 10.0.10.23/24 metric 100 brd 10.0.10.255 scope global dynamic eth0
16 valid_lft 1923sec preferred_lft 1923sec
17 inet6 fe80::6b:81ff:fe64:e132/64 scope link
18 valid_lft forever preferred_lft forever
19 3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group
20 default
21 link/ether 02:42:69:d8:e7:3d brd ff:ff:ff:ff:ff:ff
22 inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
23 valid_lft forever preferred_lft forever
24 inet6 fe80::42:69ff:fed8:e73d/64 scope link
25 valid_lft forever preferred_lft forever
26
27 $ sudo brctl show
28 bridge name bridge id STP enabled interfaces
29 docker0 8000.024269d8e73d no
30
31 $ sudo docker run --name busybox -it busybox
32 /# ifconfig
33 eth0 Link encap:Ethernet HWaddr 02:42:AC:11:00:02
34 inet addr:172.17.0.2 Bcast:172.17.255.255 Mask:255.255.0.0
35 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
36 RX packets:10 errors:0 dropped:0 overruns:0 frame:0
37 TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
38 collisions:0 txqueuelen:0
39 RX bytes:876 (876.0 B) TX bytes:0 (0.0 B)
40
41 lo Link encap:Local Loopback
42 inet addr:127.0.0.1 Mask:255.0.0.0
43 UP LOOPBACK RUNNING MTU:65536 Metric:1
44 RX packets:0 errors:0 dropped:0 overruns:0 frame:0
45 TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
46 collisions:0 txqueuelen:1000
47 RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
48
49 /# ping -c 4 8.8.8.8 <----외부 통신 가능 확인
50
51 2)자동으로 172.17.0.x의 IP Address 부여 확인하기
52 -다른 세션을 열어서
53 $ sudo docker run --name busybox1 -it busybox
54 /# ifconfig
55 eth0 Link encap:Ethernet HWaddr 02:42:AC:11:00:02
56 inet addr:172.17.0.3 Bcast:172.17.255.255 Mask:255.255.0.0
57 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
58 RX packets:9 errors:0 dropped:0 overruns:0 frame:0
59 TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
60 collisions:0 txqueuelen:0

```

```

60 RX bytes:806 (806.0 B) TX bytes:0 (0.0 B)
61
62
63 -또 다른 세션을 열어서
64 $ sudo docker run -d -p 80:80 --name web nginx
65 $ sudo docker inspect web
66 $ curl 172.17.0.4
67 $ sudo iptables -t nat -L -v
68 Chain PREROUTING (policy ACCEPT 1 packets, 84 bytes)
69   pkts bytes target     prot opt in     out     source            destination
70   815 42348 DOCKER    all  --  any    any     anywhere          anywhere          ADDRTYPE match
71   dst-type LOCAL
72 Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
73   pkts bytes target     prot opt in     out     source            destination
74
75 Chain OUTPUT (policy ACCEPT 9 packets, 1174 bytes)
76   pkts bytes target     prot opt in     out     source            destination
77   0     0 DOCKER    all  --  any    any     anywhere          !localhost/8      ADDRTYPE match
78   dst-type LOCAL
79 Chain POSTROUTING (policy ACCEPT 9 packets, 1174 bytes)
80   pkts bytes target     prot opt in     out     source            destination
81   1513 93953 MASQUERADE all  --  any    !docker0 172.17.0.0/16    anywhere
82   0     0 MASQUERADE tcp  --  any    any     172.17.0.4      172.17.0.4      tcp dpt:http
83
84 Chain DOCKER (2 references)
85   pkts bytes target     prot opt in     out     source            destination
86   0     0 RETURN    all  --  docker0 any     anywhere          anywhere
87   0     0 DNAT      tcp  --  !docker0 any     anywhere          anywhere          tcp dpt:http
88   to:172.17.0.4:80
89
90
91 2. Port-Forwarding
92 1)host의 port와 container의 port 지정해서 연결하기
93 $ sudo docker run -p 80:80 -d --name web1 nginx
94
95 $ sudo docker ps
96 CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS
97 05c359f8bcd6   nginx    "/docker-entrypoint...." About a minute ago Up About a minute
98 0.0.0.0:80->80/tcp, :::80->80/tcp    web1
99
100 $ curl localhost:80
101
102 2)host의 port를 랜덤으로 연결하기
103 $ sudo docker run -p 80 -d --name web2 nginx
104 $ sudo docker ps
105 CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS
106 8e4372270de9   nginx    "/docker-entrypoint...." 6 seconds ago    Up 5 seconds
107 0.0.0.0:49153->80/tcp, :::49153->80/tcp web2
108 05c359f8bcd6   nginx    "/docker-entrypoint...." About a minute ago Up About a minute
109 0.0.0.0:80->80/tcp, :::80->80/tcp    web1
110
111 $ curl localhost:49153
112
113 3)host와 container 모두 자동으로 연결하기
114 $ sudo docker run -P(대문자) 80 -d --name web3 nginx
115 $ sudo docker ps -a

```

	CONTAINER ID	IMAGE	COMMAND NAMES	CREATED	STATUS	
113	8ae0560aa57c	nginx	"/docker-entrypoint..."	3 seconds ago	Up 2 seconds	0.0.0.0:49154->80/tcp,
114	:::49154->80/tcp	web3				
115	8e4372270de9	nginx	"/docker-entrypoint..."	3 minutes ago	Up 3 minutes	0.0.0.0:49153->80/tcp,
	:::49153->80/tcp	web2				
116	05c359f8bcd6	nginx	"/docker-entrypoint..."	4 minutes ago	Up 4 minutes	0.0.0.0:80->80/tcp,
	:::80->80/tcp	web1				

117
118
119

120 3. user-defined network 구성하기

121 1) 기본 bridge 외에 새로 생성하기

```
122 $ sudo docker network ls
123 NETWORK ID      NAME          DRIVER  SCOPE
124 32ce6dec4771    bridge       bridge  local
125 ef8f1c31a15d    host         host    local
126 ee449dfed7eb    none        null    local
127
128 $ sudo docker network create --driver bridge --subnet 192.168.100.0/24 ₩
129 > --gateway 192.168.100.254 mynet
130 df7b218797e7216e1b39549a94ab9b0b2b5d2946be63233ed8ac1b17a62742c6
```

131

```
132 $ sudo docker network ls
133 NETWORK ID      NAME          DRIVER  SCOPE
134 32ce6dec4771    bridge       bridge  local
135 ef8f1c31a15d    host         host    local
136 df7b218797e7    mynet        bridge  local
137 ee449dfed7eb    none        null    local
```

138

139 2) 새로 생성한 bridge로 Container 생성하기

```
140 $ sudo docker run -it --name busybox1 --net mynet busybox
141 / # ifconfig
142 eth0      Link encap:Ethernet  HWaddr 02:42:C0:A8:64:01
143           inet addr:192.168.100.1  Bcast:192.168.100.255  Mask:255.255.255.0
144           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
145           RX packets:14 errors:0 dropped:0 overruns:0 frame:0
146           TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
147           collisions:0 txqueuelen:0
148           RX bytes:1252 (1.2 KiB)  TX bytes:0 (0.0 B)
```

149

```
150 lo        Link encap:Local Loopback
151           inet addr:127.0.0.1  Mask:255.0.0.0
152           UP LOOPBACK RUNNING  MTU:65536  Metric:1
153           RX packets:0 errors:0 dropped:0 overruns:0 frame:0
154           TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
155           collisions:0 txqueuelen:1000
156           RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

```
157 /# exit
```

158

159

```
160 $ sudo docker inspect mynet
```

```
161 [
162     {
163         "Name": "mynet",
164         "Id": "df7b218797e7216e1b39549a94ab9b0b2b5d2946be63233ed8ac1b17a62742c6",
165         "Created": "2021-10-21T08:59:00.152346729Z",
166         "Scope": "local",
167         "Driver": "bridge",
168         "EnableIPv6": false,
169         "IPAM": {
```

```

170         "Driver": "default",
171         "Options": {},
172         "Config": [
173             {
174                 "Subnet": "192.168.100.0/24",
175                 "Gateway": "192.168.100.254"
176             }
177         ]
178     },
179     "Internal": false,
180     "Attachable": false,
181     "Ingress": false,
182     "ConfigFrom": {
183         "Network": ""
184     },
185     "ConfigOnly": false,
186     "Containers": {},
187     "Options": {},
188     "Labels": {}
189 }
190 ]
191
192

```

3) Container 생성시 ip 지정하기

```

194 $ sudo docker run -it --name busybox2 --net mynet --ip 192.168.100.100 busybox
195 / # ifconfig
196 eth0      Link encap:Ethernet  HWaddr 02:42:C0:A8:64:64
197           inet addr:192.168.100.100  Bcast:192.168.100.255  Mask:255.255.255.0
198           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
199           RX packets:8 errors:0 dropped:0 overruns:0 frame:0
200           TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
201           collisions:0 txqueuelen:0
202           RX bytes:736 (736.0 B)  TX bytes:0 (0.0 B)
203
204 lo        Link encap:Local Loopback
205           inet addr:127.0.0.1  Mask:255.0.0.0
206           UP LOOPBACK RUNNING  MTU:65536  Metric:1
207           RX packets:0 errors:0 dropped:0 overruns:0 frame:0
208           TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
209           collisions:0 txqueuelen:1000
210           RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
211
212 / # ping -c 4 8.8.8.8
213
214
215

```

4. Container 간 통신하기

1) 첫번째 방법

-MySQL 실행하기

```

219 $ docker run -d -p 3306:3306 ₩
220 > -e MYSQL_ALLOW_EMPTY_PASSWORD=true ₩
221 > --name mysql ₩
222 > mysql:5.7
223
224 $ docker exec -it mysql mysql
225 mysql> CREATE DATABASE wp CHARACTER SET utf8;
226 mysql> GRANT ALL PRIVILEGES ON wp.* TO 'wp'@'%' IDENTIFIED BY 'wp';
227 mysql> FLUSH PRIVILEGES;
228 mysql> show databases;
229 +-----+
230 | Database          |

```

```

231 +-----+
232 | information_schema |
233 | mysql              |
234 | performance_schema |
235 | sys                |
236 | wp                  |
237 +-----+
238 5 rows in set (0.00 sec)
239 mysql> quit
240

```

241 -WordPress 실행하기

```

242 $ docker run -d -p 8080:80 ₩
243 > -e WORDPRESS_DB_HOST=host.docker.internal ₩ <---Linux에서는 연결안됨. WSL만 가능
244 > -e WORDPRESS_DB_NAME=wp ₩
245 > -e WORDPRESS_DB_USER=wp ₩
246 > -e WORDPRESS_DB_PASSWORD=wp ₩
247 > --name wordpress
248 > wordpress
249

```

250 -브라우저에서 연결

```

251 http://localhost:8080
252
253

```

254 2)두번째 방법

255 -MySQL 실행하기

```

256 $ docker run -d -p 3306:3306 ₩
257 > -e MYSQL_ALLOW_EMPTY_PASSWORD=true ₩
258 > --name mysql ₩
259 > mysql:5.7
260

```

261 -MySQL에 wp 데이터베이스 생성 및 wp 계정 생성

```

262 $ docker exec -it mysql mysql
263 mysql> CREATE DATABASE wp CHARACTER SET utf8;
264 mysql> GRANT ALL PRIVILEGES ON wp.* TO 'wp'@'%' IDENTIFIED BY 'wp';
265 mysql> FLUSH PRIVILEGES;
266 mysql> show databases;
267 +-----+
268 | Database          |
269 +-----+
270 | information_schema |
271 | mysql              |
272 | performance_schema |
273 | sys                |
274 | wp                  |
275 +-----+
276 5 rows in set (0.00 sec)
277 mysql> quit
278

```

279 -app-network 라는 이름으로 wordpress와 MySQL이 통신할 네트워크 만들기

```

280 $ docker network create app-network
281

```

282 -MySQL containier에 네트워크를 추가

```

283 $ docker network connect app-network mysql
284

```

285 -network option 사용하기

286 -WordPress를 app-network에 속하게 하고 mysql을 이름으로 접근한다.

```

287 $ docker run -dp 8080:80 ₩
288 > --network=app-network ₩
289 > -e WORDPRESS_DB_HOST=mysql ₩
290 > -e WORDPRESS_DB_NAME=wp ₩
291 > -e WORDPRESS_DB_USER=wp ₩

```

```
292 > -e WORDPRESS_DB_PASSWORD=wp ₩
293 > wordpress
```

```
294
295 -웹 브라우저에서 확인
296 http://HOST-IP:8080
```

3)세번째 방법

```
300 - wordpress와 mysql 컨테이너 삭제
301   $ sudo docker rm -f `docker ps -a -q`
302   $ sudo docker ps -a
303   $ sudo docker rmi `docker images -q`
```

```
304
305 -/dbdata 디렉토리 삭제
306   $ sudo rm -rf /dbdata
```

```
307
308 -MySQL 실행하기
309   $ sudo docker run -d -p 3306:3306 ₩
310   > --name mysql -v /dbdata:/var/lib/mysql ₩
311   > -e MYSQL_ROOT_PASSWORD=wordpress ₩
312   > -e MYSQL_PASSWORD=wordpress mysql:5.7
```

```
313
314 -MySQL에 wp 데이터베이스 생성 및 wp 계정 생성
315   $ docker exec -it mysql bash
316   bash-4.2# mysql -h localhost -u root -p
317   Enter password:wordpress
318   mysql> CREATE DATABASE wp CHARACTER SET utf8;
319   mysql> GRANT ALL PRIVILEGES ON wp.* TO 'wp'@'%' IDENTIFIED BY 'wp';
320   mysql> FLUSH PRIVILEGES;
321   mysql> show databases;
322   +-----+
323   | Database          |
324   +-----+
325   | information_schema |
326   | mysql              |
327   | performance_schema |
328   | sys                |
329   | wp                  |
330   +-----+
331   5 rows in set (0.00 sec)
332   mysql> quit
333   bash-4.2# exit
334   exit
```

```
335
336 $ sudo docker ps -a
```

```
337
338 -wordpress container 실행
339   $ sudo docker run -dp 8080:80 ₩
340   > --name wordpress --link mysql:mymysql ₩ <--link의 이름의 앞부분은 mysql의 Container의 이름,
341   뒷부분은 자유
342   > -e WORDPRESS_DB_PASSWORD=wordpress ₩
343   > -e WORDPRESS_DB_HOST=mysql ₩
344   > -e WORDPRESS_DB_NAME=wp ₩
345   > -e WORDPRESS_DB_USER=wp ₩
346   > wordpress
```

```
347 $ sudo docker ps -a
```

```
348
349 -웹 브라우저에서 확인
350 http://HOST-IP:8080
```

```
351
```

