**IPvlan vs. MacVLAN**

* The MacVLAN network allocates a unique MAC address to every container. Thus, a single network interface on a Docker host essentially advertises multiple MAC addresses. With an IPvlan network, all containers on a Docker host share a single MAC address.
* A MacVLAN network requires you to enable promiscuous mode on the parent interface of the Docker host, which is not required with IPvlan.
* IPvlan can work in two modes: L2 mode and L3 mode.
* L2 (or Layer 2) mode is the default mode for IPvlan. It works just like MacVLAN (bridge) mode but without assigning a unique MAC address to each container. In L2 mode, the Docker host acts like a switch between the parent interface and a virtual NIC for each container. The entire communication is based only on MAC addresses. The containers in one IPvlan network (in L2 mode) can communicate with the containers in another IPvlan network. However, this generates many ARP broadcasts, which affects network performance.
* In L3 (or Layer 3) mode, the Docker host works like a Layer 3 device to route the packets between the parent interface and the virtual NIC for each container. In this mode, containers are totally isolated from other networks, and the broadcasts are restricted to the Layer 2 subnet only; this improves network performance. The only downside is that you need to manually add a static route on your gateway router to let other network devices know how to reach your IPvlan network running in L3 mode.

You just need to remember that you cannot create multiple IPvlan networks with one parent interface on the Docker host. One parent interface can serve only one IPvlan network. If you use one parent interface and define multiple subnets, the containers of both subnets will be able to communicate with each other, by default, without any additional configuration.

docker network create --driver ipvlan --subnet 192.168.1.0/24 --gateway 192.168.1.1 --opt parent=ens33 --opt ipvlan\_mode=l2 nhipvlan

docker run -itd --name web --rm --ip 192.168.1.60 --network nhipvlan alpine

docker run -itd --name database --rm --ip 192.168.1.61 --network nhipvlan --env MARIADB\_ROOT\_PASSWORD=foobar mariadb

# NOTE: the containers can NOT ping the underlying host interfaces as  
# they are intentionally filtered by Linux for additional isolation.

*Host network:*

docker run -it --name weba --network=host alpine

docker run -it --name webab --network=none alpine

<https://github.com/XP2600-hub/nhorizon-python-dcompose.git>

**Memory and CPU limits:**

docker run -dit --memory="[memory-limit]" [docker-image]

docker run -dit --memory="256m" nginx

docker info

vim /etc/default/grub

GRUB\_CMDLINE\_LINUX="cdgroup\_enable=memory swapaccount=1"

update-grub

docker run -dit --memory="512m" --memory-swap="1g" nginx

**Soft Limit to Container Memory**

docker run -dit --memory="1g" --memory-reservation="512m" nginx

docker stats

**Limit Number of CPU Cores**

docker run -dit --cpus="1.0" nginx

**Limit CPU Cycles**

docker run -dit --cpu-shares="2048" nginx

***Default value 1024***