**IPvlan Network**

IPvlan (IP Virtual LAN) is a Linux kernel feature that allows you to create multiple virtual network interfaces with their own MAC addresses on a single physical network interface. This enables you to assign unique IP addresses to each virtual interface, effectively creating multiple IP subnets on the same physical network. Docker leverages this feature to provide the IPvlan network driver, which allows containers to communicate directly with the underlying physical network.

Advantages of Docker IPvlan Network:

* Performance: IPvlan provides better performance compared to some other Docker network drivers because it operates at the kernel level. Since there's less overhead in user-space, the network operations can be more efficient.
* Layer 2 Isolation: Containers attached to the same IPvlan network are isolated at Layer 2, which means they operate as if they are on separate physical networks. This isolation can enhance security by preventing direct communication between containers.
* Host Integration: Containers connected to an IPvlan network can communicate with the host machine and other devices on the physical network. This is useful for scenarios where you want containers to be directly reachable from other machines on the network.
* No Overlay Network Overhead: Unlike some other Docker network drivers (e.g., overlay networks), IPvlan doesn't add additional encapsulation or overhead to the network traffic. This simplicity can be beneficial in certain situations.

Disadvantages of Docker IPvlan Network:

* Kernel Version Dependency: IPvlan support depends on the version of the Linux kernel. In some environments, you may encounter compatibility issues if the kernel version is not appropriate.
* Limited to Linux Hosts: IPvlan is a Linux-specific feature, so it cannot be used on non-Linux hosts. If you have a heterogeneous environment with multiple operating systems, you might need to consider other Docker network drivers that are more cross-platform.
* Complex Configuration: Setting up and configuring IPvlan networks can be more complex compared to some other Docker network drivers. This might be a disadvantage for users who are new to networking concepts.
* Limited Docker Features: IPvlan does not support all Docker features, such as multicast or broadcast. If your application relies on these features, you may need to choose a different network driver.

In summary, Docker IPvlan networks offer performance benefits and layer 2 isolation but come with potential complexity and limitations, making them suitable for certain use cases and environments. It's important to consider your specific requirements and the characteristics of your network when choosing a Docker network driver.

Let’s do practical:-

Docker IPVLAN networking allows you to connect containers directly to your physical network, each with its own MAC and IP addresses. This provides better performance and network isolation. Here's a brief explanation and example:

1. Create a Docker network with IPVLAN driver:

*docker network create -d ipvlan --subnet=192.168.1.0/24 --gateway=192.168.1.1 -o parent=eth0 my\_ipvlan\_network*

*(*Replace eth0 with your host's network interface.)

1. Run containers on the created network:

*docker run -d --network=my\_ipvlan\_network --name container1 my\_image*

*docker run -d --network=my\_ipvlan\_network --name container2 my\_image*

(Now, container1 and container2 are connected to the same IPVLAN network.)

1. Check the IP addresses:

*docker exec container1 ip addr show eth0*

*docker exec container2 ip addr show eth0*

(Each container will have its own IP address within the specified subnet.)

1. Verify connectivity:

*docker exec container1 ping <container2\_IP>*

(Ensure that the containers can communicate with each other.)

Remember, this is a basic example, and you might need to adjust the subnet, gateway, and other parameters based on your network configuration.

**NONE**

Docker doesn't explicitly have a "none" network as a predefined network type. The "none" network in the context above is more of an option you specify when running a container, indicating that the container should not be connected to any network.

If you specify the --network none option when running a container, it means the container will not be connected to any network. This essentially disables networking for the container.

Use case:

Networkless Containers: There might be situations where you don't want a container to have any network connectivity. This could be useful in scenarios where the container performs some local processing without the need for external communication.

To create none network :

*docker run --network none my\_container*

Ensure to check the latest Docker documentation for any updates or changes to network-related features, as the platform evolves over time.