IPv6 & Containers

Pieter Lewyllie

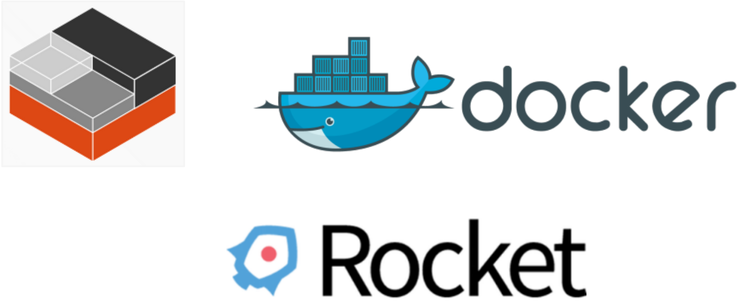
Systems Engineer @ Cisco

Agenda

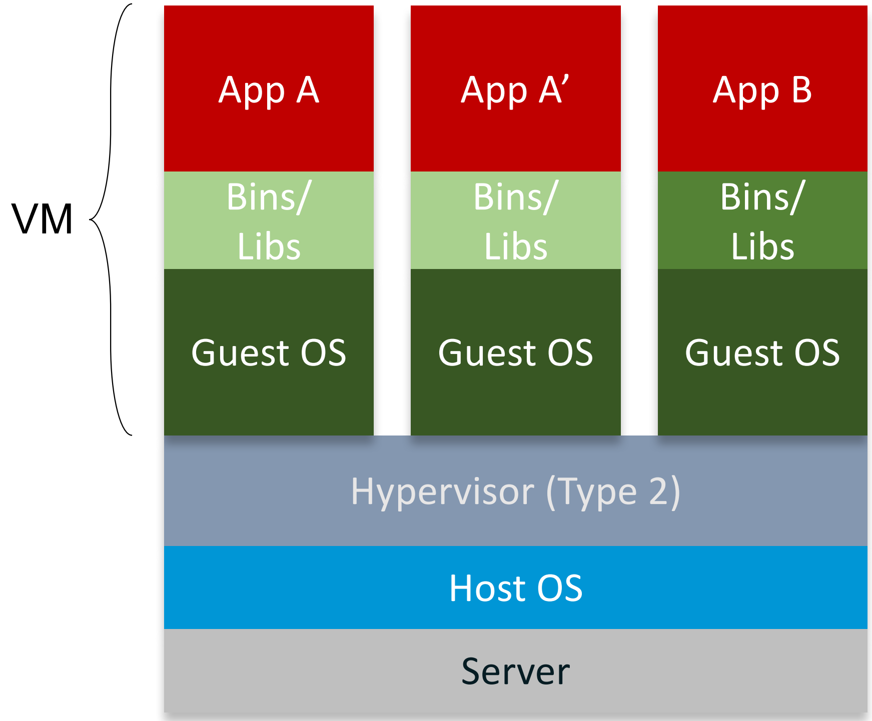
- Containers?
- IPv6 for Docker
- IPv6 for Kubernetes

Linux Containers

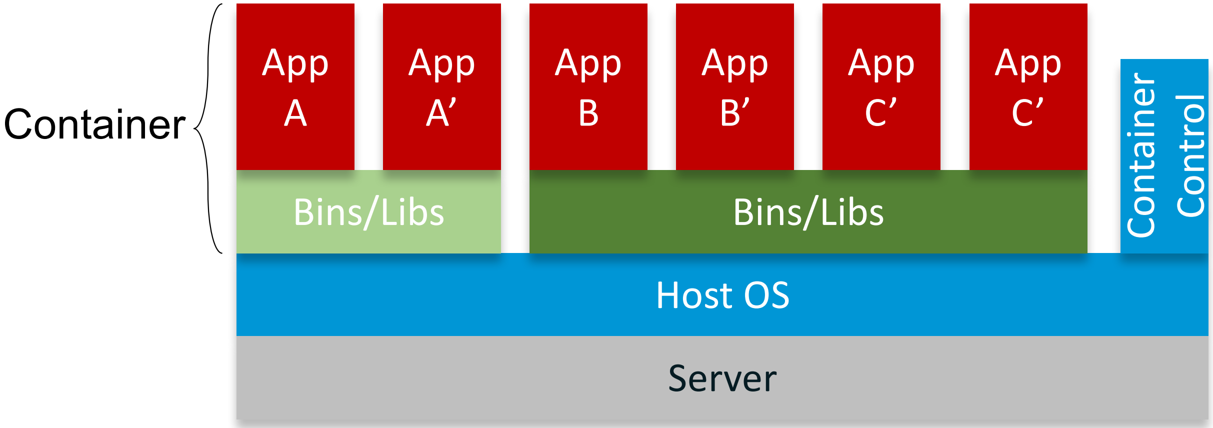
- A Linux container lets you run a Linux system within another Linux system.
- A container is a group of processes on a Linux machine.
- Those processes form an isolated environment.
- Inside the container, it (almost) looks like a VM.
- Outside the container, it looks like normal processes running on the machine.
- It looks like a VM, but it is more efficient: Containers = Lightweight Virtualization



Containers and Virtual Machines



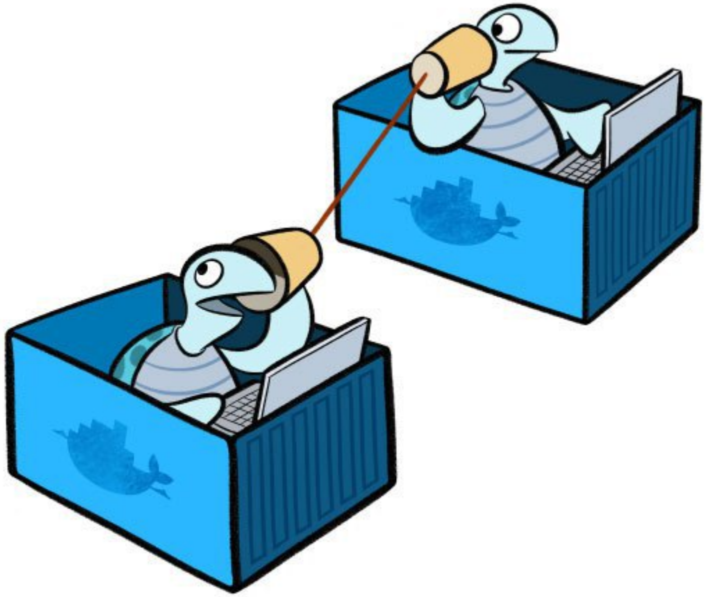
Containers are isolated but share OS and where appropriate bins/libraries



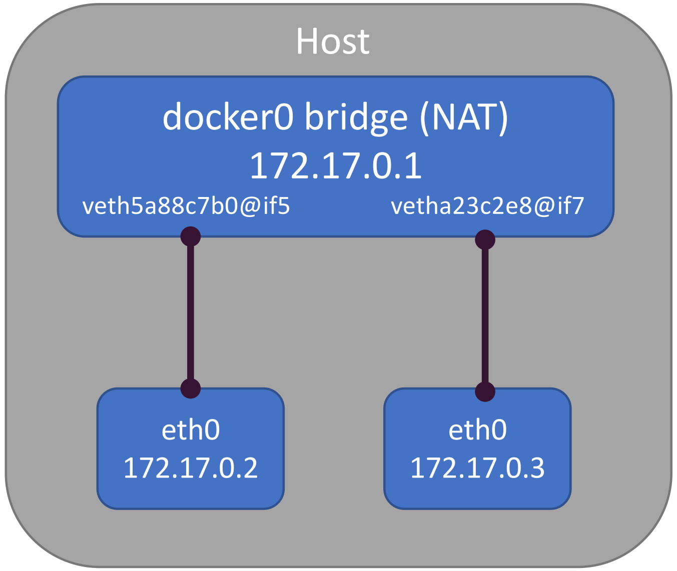
Containers are almost like Virtual Machines

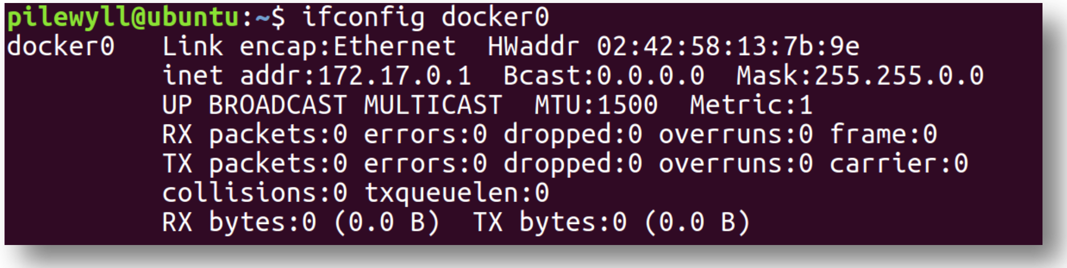
- Containers have their own network interface (and IP address)
  - Can be bridged, routed... just like with KVM, VMware etc.
- Containers have their own filesystem
  - For example a Debian host can run Fedora container (and vice-versa)
- Security: Containers are isolated from each other
  - Two containers can't harm (or even see) each other
- Resource Control: Containers are isolated and can have dedicated resources
  - Soft & hard quotas for RAM, CPU, I/O...
- Though…
- Apps in Containers share the kernel of the host OS (i.e. Linux guests only)
- Containers are light-weight, fast to start, allow for >10x density compared to VMs

Docker networking

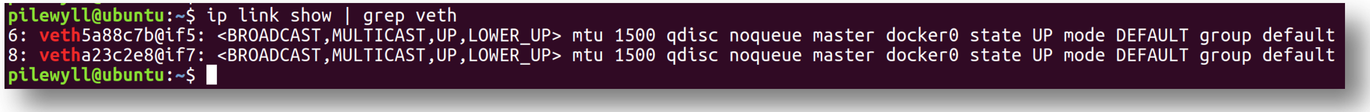


Docker Containers are connected using a bridge





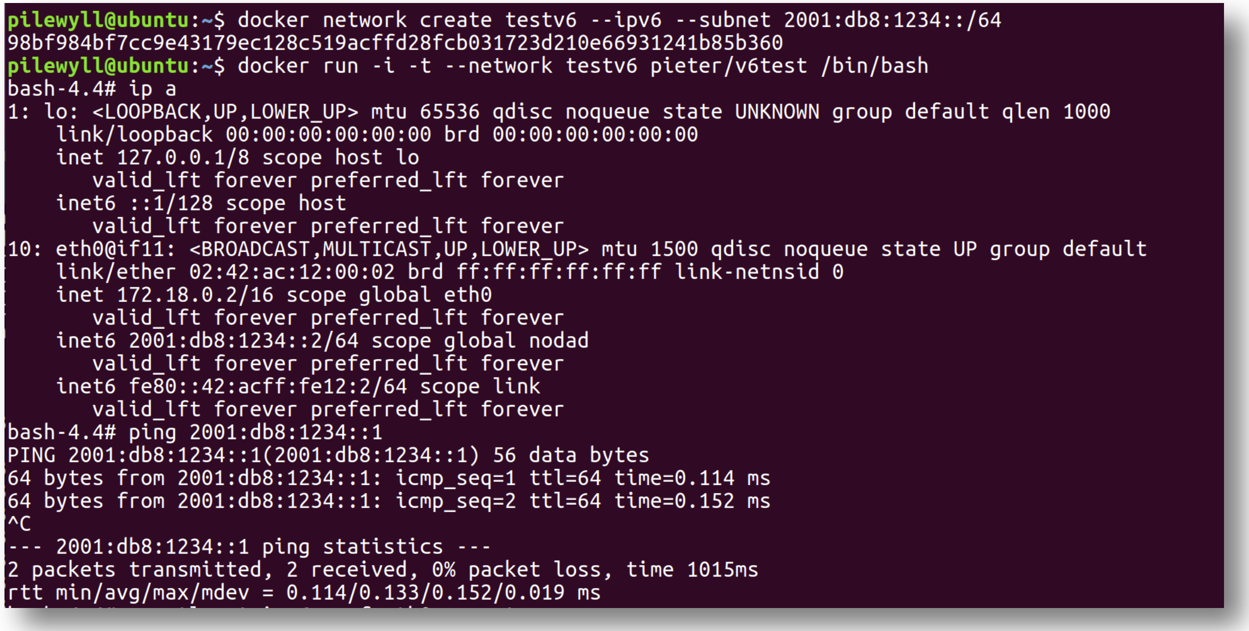
```
pilewyll@ubuntu:~$ ifconfig docker0
docker0   Link encap:Ethernet  HWaddr 02:42:58:13:7b:9e
          inet addr:172.17.0.1  Bcast:0.0.0.0  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```



```
pilewyll@ubuntu:~$ ip link show | grep veth
6: veth5a88c7b@if5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP mode DEFAULT group default
8: vetha23c2e8@if7: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP mode DEFAULT group default
pilewyll@ubuntu:~$
```

Docker bridge with IPv6

- No more NAT, all ports are exposed
- Docker assigns IPv6 addresses sequentially
- Default GW needs to be in same subnet
- Set accept_ra to 2

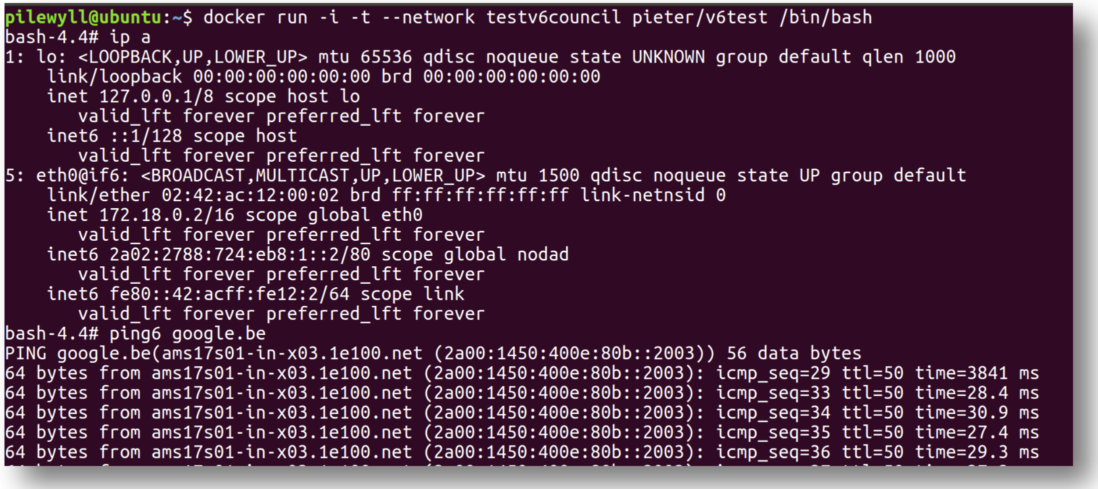


```
pilewyll@ubuntu:~$ docker network create testv6 --ipv6 --subnet 2001:db8:1234::/64
98bf984bf7cc9e43179ec128c519acffd28fcb031723d210e66931241b85b360
pilewyll@ubuntu:~$ docker run -i -t --network testv6 pieter/v6test /bin/bash
bash-4.4# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
10: eth0@if11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.2/16 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1234::2/64 scope global nodad
       valid_lft forever preferred_lft forever
    inet6 fe80::42:acff:fe12:2/64 scope link
       valid_lft forever preferred_lft forever
bash-4.4# ping 2001:db8:1234::1
PING 2001:db8:1234::1(2001:db8:1234::1) 56 data bytes
64 bytes from 2001:db8:1234::1: icmp_seq=1 ttl=64 time=0.114 ms
64 bytes from 2001:db8:1234::1: icmp_seq=2 ttl=64 time=0.152 ms
^C
--- 2001:db8:1234::1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1015ms
rtt min/avg/max/mdev = 0.114/0.133/0.152/0.019 ms
```

https://docs.docker.com/v17.09/engine/userguide/networking/default_network/ipv6

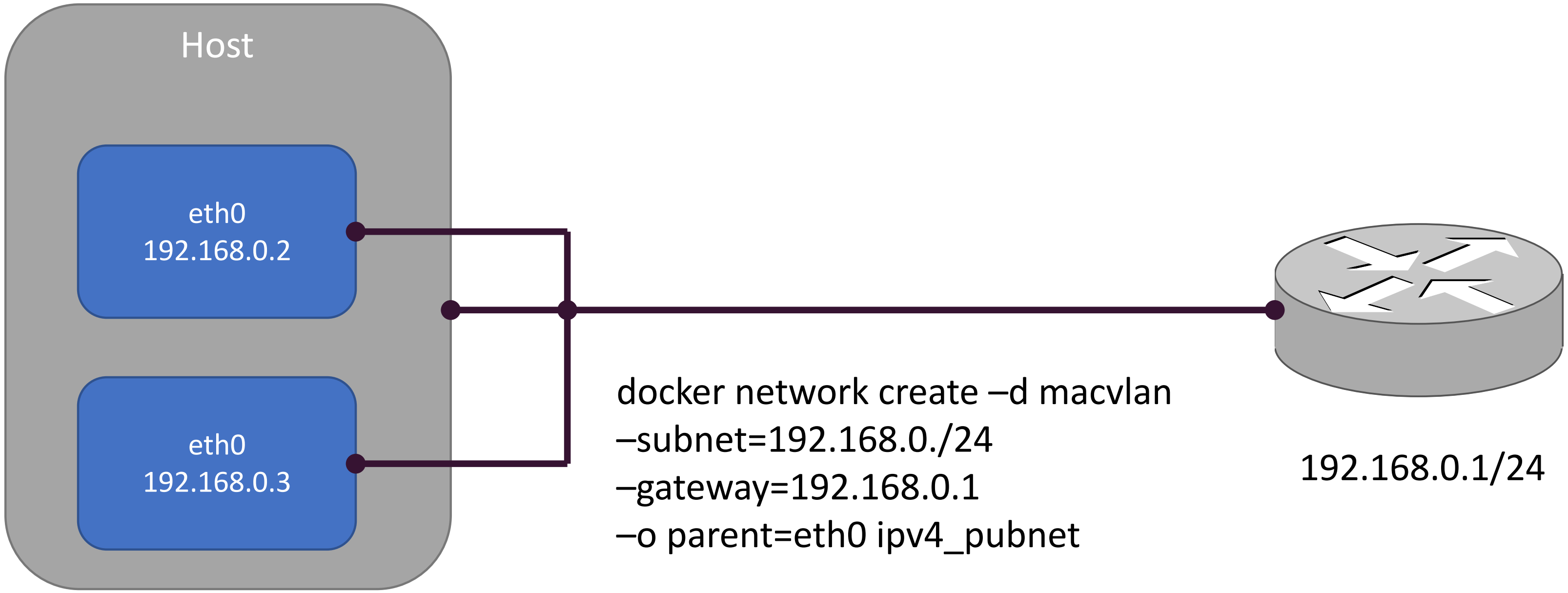
Reaching the internet with NDP Proxy

- sudo sysctl net.ipv6.conf.default.proxy_ndp=1
- sudo sysctl net.ipv6.conf.all.proxy_ndp=1
- docker network create testv6council --ipv6 --subnet 2a02:2789:724:eb8:1::/80
- sudo ip -6 neigh add proxy 2a02:2789:724:eb8:1:ff:ff:ff dev br-e25957a13b1f
- sudo ip -6 neigh add proxy 2a02:2789:724:eb8:1::2 dev ens33



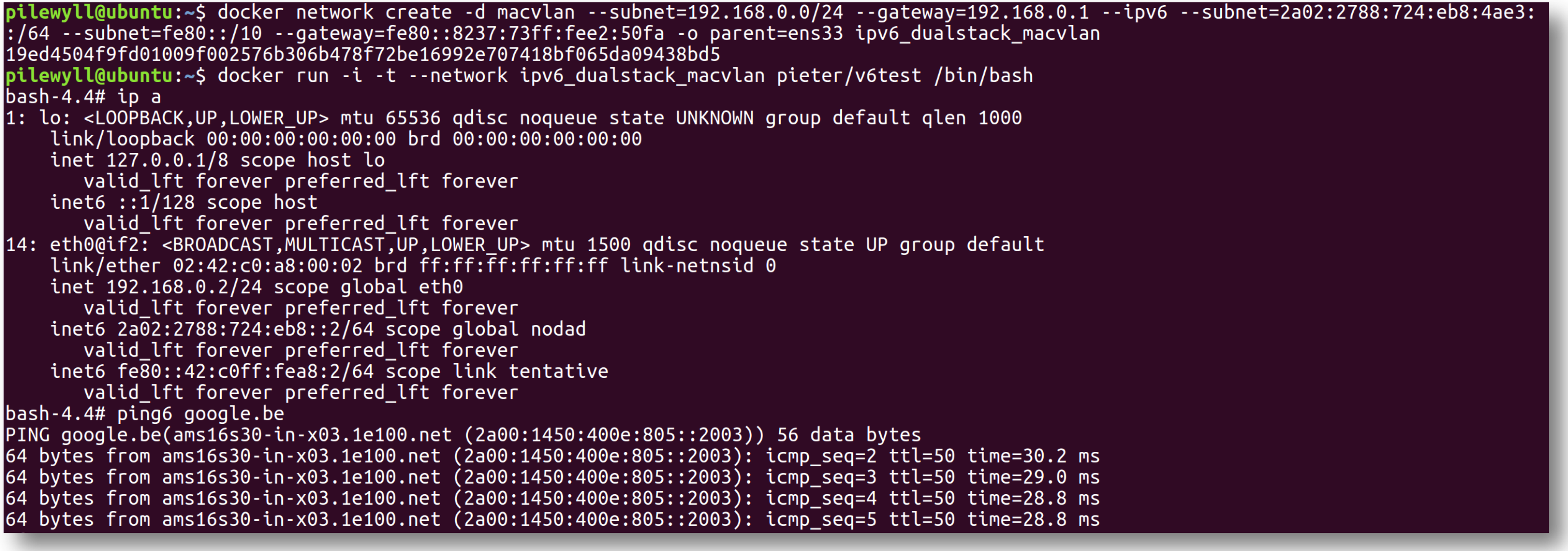
```
pilewyll@ubuntu:~$ docker run -i -t --network testv6council pieter/v6test /bin/bash
bash-4.4# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
5: eth0@if6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.2/16 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 2a02:2788:724:eb8:1::2/80 scope global nodad
       valid_lft forever preferred_lft forever
    inet6 fe80::42:acff:fe12:2/64 scope link
       valid_lft forever preferred_lft forever
bash-4.4# ping6 google.be
PING google.be(ams17s01-in-x03.1e100.net (2a00:1450:400e:80b::2003)) 56 data bytes
64 bytes from ams17s01-in-x03.1e100.net (2a00:1450:400e:80b::2003): icmp_seq=29 ttl=50 time=3841 ms
64 bytes from ams17s01-in-x03.1e100.net (2a00:1450:400e:80b::2003): icmp_seq=33 ttl=50 time=28.4 ms
64 bytes from ams17s01-in-x03.1e100.net (2a00:1450:400e:80b::2003): icmp_seq=34 ttl=50 time=30.9 ms
64 bytes from ams17s01-in-x03.1e100.net (2a00:1450:400e:80b::2003): icmp_seq=35 ttl=50 time=27.4 ms
64 bytes from ams17s01-in-x03.1e100.net (2a00:1450:400e:80b::2003): icmp_seq=36 ttl=50 time=29.3 ms
```

Macvlan



```
docker network create –d macvlan
–subnet=192.168.0./24
–gateway=192.168.0.1
–o parent=eth0 ipv4_pubnet
```

IPv6 Macvlan



```
pilewyll@ubuntu:~$ docker network create -d macvlan --subnet=192.168.0.0/24 --gateway=192.168.0.1 --ipv6 --subnet=2a02:2788:724:eb8:4ae3:
:/64 --subnet=fe80::/10 --gateway=fe80::8237:73ff:fee2:50fa -o parent=ens33 ipv6_dualstack_macvlan
19ed4504f9fd01009f002576b306b478f72be16992e707418bf065da09438bd5
pilewyll@ubuntu:~$ docker run -i -t --network ipv6_dualstack_macvlan pieter/v6test /bin/bash
bash-4.4# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
14: eth0@if2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:a8:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.0.2/24 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 2a02:2788:724:eb8::2/64 scope global nodad
       valid_lft forever preferred_lft forever
    inet6 fe80::42:c0ff:fea8:2/64 scope link tentative
       valid_lft forever preferred_lft forever
bash-4.4# ping6 google.be
PING google.be(ams16s30-in-x03.1e100.net (2a00:1450:400e:805::2003)) 56 data bytes
64 bytes from ams16s30-in-x03.1e100.net (2a00:1450:400e:805::2003): icmp_seq=2 ttl=50 time=30.2 ms
64 bytes from ams16s30-in-x03.1e100.net (2a00:1450:400e:805::2003): icmp_seq=3 ttl=50 time=29.0 ms
64 bytes from ams16s30-in-x03.1e100.net (2a00:1450:400e:805::2003): icmp_seq=4 ttl=50 time=28.8 ms
64 bytes from ams16s30-in-x03.1e100.net (2a00:1450:400e:805::2003): icmp_seq=5 ttl=50 time=28.8 ms
```

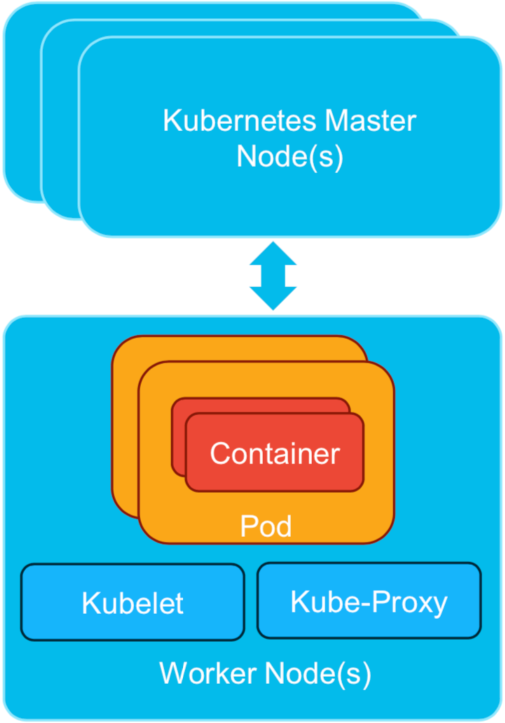
https://docs.docker.com/network/macvlan

Docker Networking

- Bridge network driver (--driver=bridge)
    - IPv6 can be enabled (--ipv6)
- None network driver (--driver=none)
- Host network driver (--driver=host)
- Overlay network driver (--driver=overlay) – Multi-Host using VXLAN
- MACVLAN network driver (--driver=macvlan)
    - IPv6 can be enabled
- Remote drivers – compatible with CNM (Container Network Model)
    - Contiv, Weave, Calico…

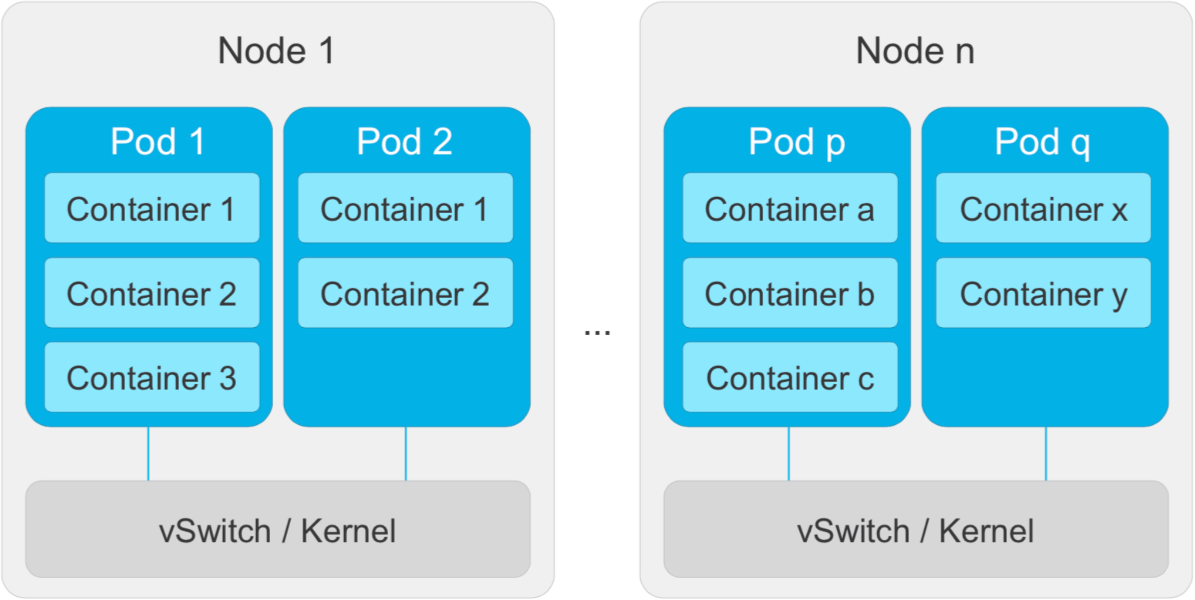
Kubernetes

- Container orchestrator
- Runs and manages containers
- Supports multiple cloud and bare-metal environments
- Inspired and informed by Google's experiences and internal systems
- 100% Open source, written in Go
- Manage applications, not machines
- Rich ecosystem of plug-ins for scheduling, storage, networking



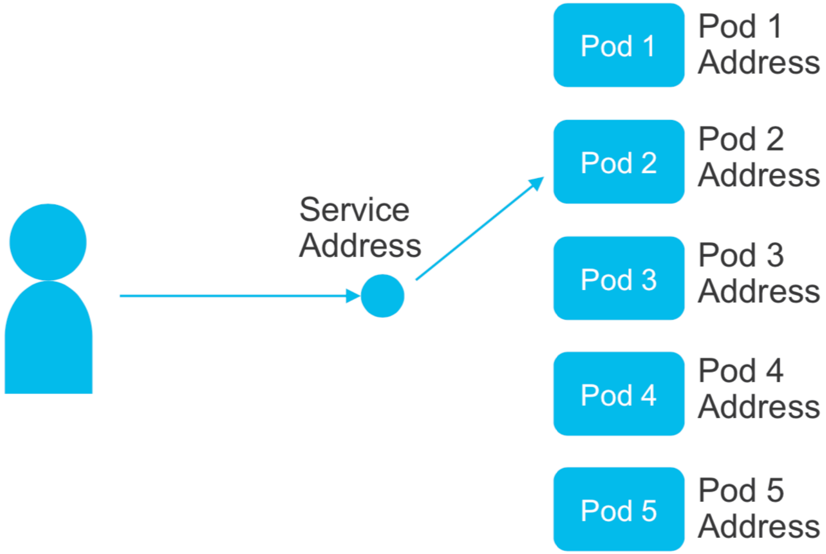
Nodes, Pods, Containers

- Node:
  - A server
- Cluster:
  - Collection of nodes
- Pod:
  - Collection of containers;
  - Nodes can run multiple Pods



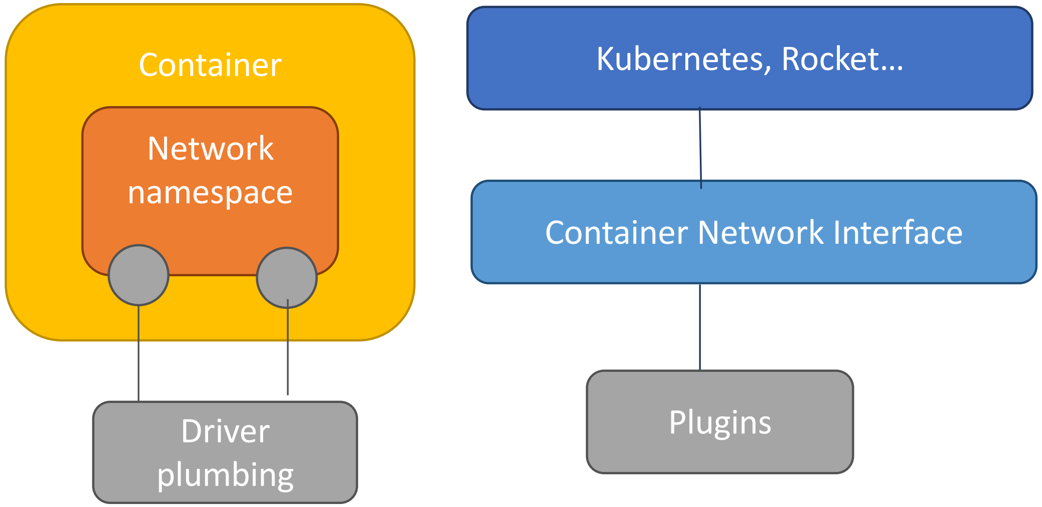
Services overview

- “Pods can come and go, services stay”
- Define a single IP/Port combination that provides access to a pool of pods
- By default a service connects the client to a Pod in a round- robin fashion
- This solves the dilemma of having to keep up with every transient IP address assigned by Docker



Container Network Interface (CNI)

- Proposed by CoreOS as part of appc specification
- Common interface between container run time and network plugin
- Gives driver freedom to manipulate network namespace
- Network described by JSON config
- Plugins support two commands:
  - Add Container to Network
  - Remove Container from Network
- Many CNI plugins available:
  - Calico, Flannel, Weave, Contiv…



IPv6 in Kubernetes

- IPv4 Parity, no API Changes
- CNI 0.6.0 Bridge & Host-Local IPAM
- ip6tables & ipvs
- Kube-DNS & CoreDNS
- kubeadm

Rel 1.9 (Alpha)

Rel 1.11 (Beta)

- Dual-Stack, parallel IPv4/IPv6
- Multiple IPs per pod
- Multiple IPs per service

Rel 1.12 (targeting)

- SRv6
- Istio IPv6
- Multiprefix Routing…

Planning and Preparing

Source: SRv6LB @ Kubecon https://www.youtube.com/watch?v=RRKUeyFaqEA

IPv6 Containers @ Facebook (!k8s)

- Every server gets a /64
- Unique IPv6 Address per *task*
  - Each task gets its own IPv6 /128
  - Each task gets the entire port space
  - No more port collisions (!!!)
  - Simpler scheduling and accounting
- /54 per Rack
- /44 per Cluster (/48 in edge)
- /37 DC Fabric
- No NATs!

What about the public cloud?

- GCE/GKE does not have IPv6 support
  - VPC networks only support IPv4 unicast traffic. They do not support broadcast, multicast, or IPv6 traffic within the network.
  - Can use IPv6 with load-balancing:
    - https://cloud.google.com/compute/docs/load-balancing/ipv6
- Azure, no IPv6 on AKS
  - IPv6 load-balancer:
    - https://docs.microsoft.com/en-us/azure/load-balancer/load-balancer-ipv6-overview
  - Long list of limitations:
    - A single IPv6 address can be assigned to a single network interface in each VM.
    - The load balancer routes the IPv6 packets to the private IPv6 addresses of the VMs using network address translation (NAT).
    - Azure VMs cannot connect over IPv6 to other VMs, other Azure services, or on-premises devices. They can only communicate with the Azure load balancer over IPv6. However, they can communicate with these other resources using IPv4.
- Amazon
  - Should work with EC2 instances
  - Each VPC is given a unique /56 address prefix from within Amazon's GUA (Global Unicast Address); you can assign a /64 address prefix to each subnet in your VPC
  - Maximum amount of IPv6 addresses per interface: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-eni.html#AvailableIpPerENI

Why IPv6 in containers?

- Future-ready, IPv6 is coming anyway…
- Less configuration (no port forwarding)
- Less state (no remembering which port for which service)
- Less moving parts (easier diagnosis of faults)
- Less variation between deployments
- Forces you to do proper security

The scale of IPv6 for containers

- Every docker host a routed /64
- **Never** re-use IPv6 address again
- How long would it take to burn through that /64?
- How about 10,000,000 per second ?
- A standard /64 prefix in IPv6 is 18,446,744,073,709,600,000 addresses.
- 18,446,744,073,709,600,000 IPv6 addresses / (10,000,000 IPv6 addresses/second * 60 sec/min * 60 min/hr * 24 hr/day * 365 days/yr) = 58,494 years
- A single /48 contains 65536 /64s
- 58,494 years * 65536 = 3,833,478,626 (3.8 *billion* years)

Ed Horley (VP engineering Groupware)
http://www.howfunky.com/2015/06/ipv6-docker-and-building-for-scale.html

References

- IPv6 and containers – a horror story
  - Matt Palmer (Linux bearded guy)
  - https://blog.apnic.net/2018/03/22/ipv6-and-containers-a-horror-story/
- SRv6LB @ Kubecon
  - Pierre Pfister (Cisco SE) & Mark Townsley (Cisco Fellow)
  - https://www.youtube.com/watch?v=RRKUeyFaqEA
- BRKSDN-2115
  - Frank Brockners (Cisco Distinguished Engineer)
  - https://www.ciscolive.com/global/on-demand-library/?search=BRKSDN-2115#/session/BRKSDN-2115
- Containers, virtualisation and IPv6
  - Steve Youell (JP Morgan)
  - http://www.ipv6.org.uk/2016/08/31/ipv6-council-meeting-october-2016/
- IPv6 in cloud deployments
  - Shannon McFarland (Cisco Distinguished Engineer)
  - http://www.rmv6tf.org/wp-content/uploads/2017/04/04-IPv6-Cloud-Deployment-RMv6tf-submit-min-1.pdf
- IPv6, Docker and building for scale
  - Ed Horley (Groupware)
  - http://www.howfunky.com/2015/06/ipv6-docker-and-building-for-scale.html

Thanks!