Containers Deep Dive

Mohammad Karimi

We’re going to talk about:

- Container (from outside)
- Linux lies!
  - Process
  - Virtual Memory
  - Process management
  - Namespaces
  - CGroup
- Container Runtimes

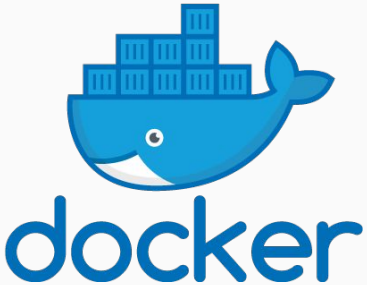
Containers

(From Outside)

Containers (from outside)

- We know containers for being
  - Lightweight
  - Portable
  - Secure

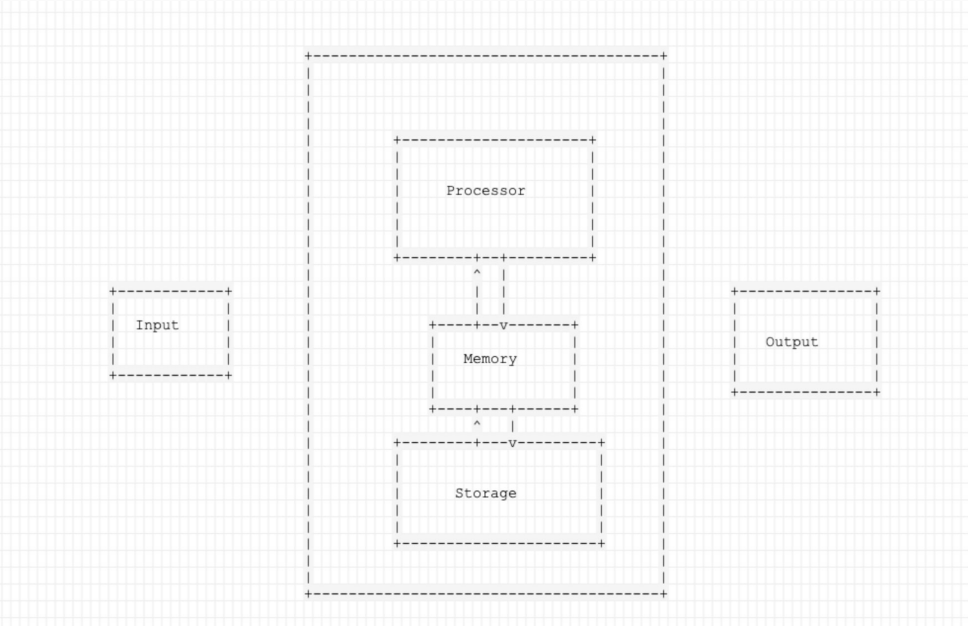
- Is there a hypervisor ?
- Is it running on host kernel ?
- Container vs VM



Linux Lies!

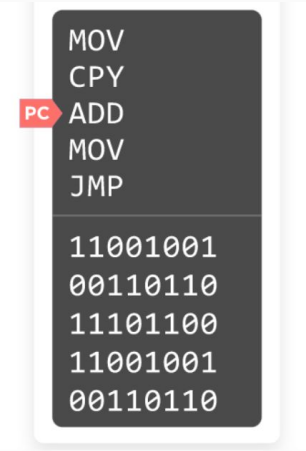
Process

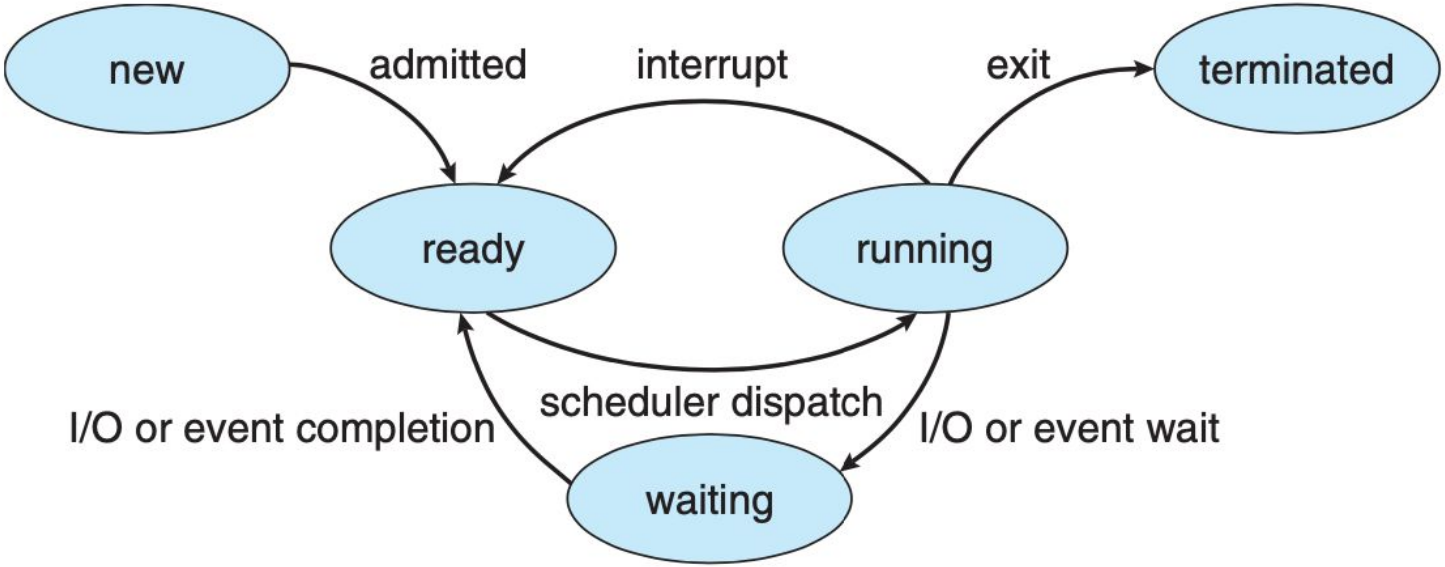
- Generally what computer does
- Run a series of instructions
- Long series of instruction:
  program



Process

- First program to run : Operating System
- Lots of processes and limited resources!
- Scheduler!
- OS Keeps track of process states
- Context Switch
- Other processes are communicating with OS




new
admitted
interrupt
exit
terminated
ready
running
scheduler dispatch
I/O or event completion
I/O or event wait
waiting

Virtual Memory

- A process thinks he’s alone!
- Process (virtual) memory starts at 0
- OS translates process-local memory into physical memory
- Simplicity and security

Process Management

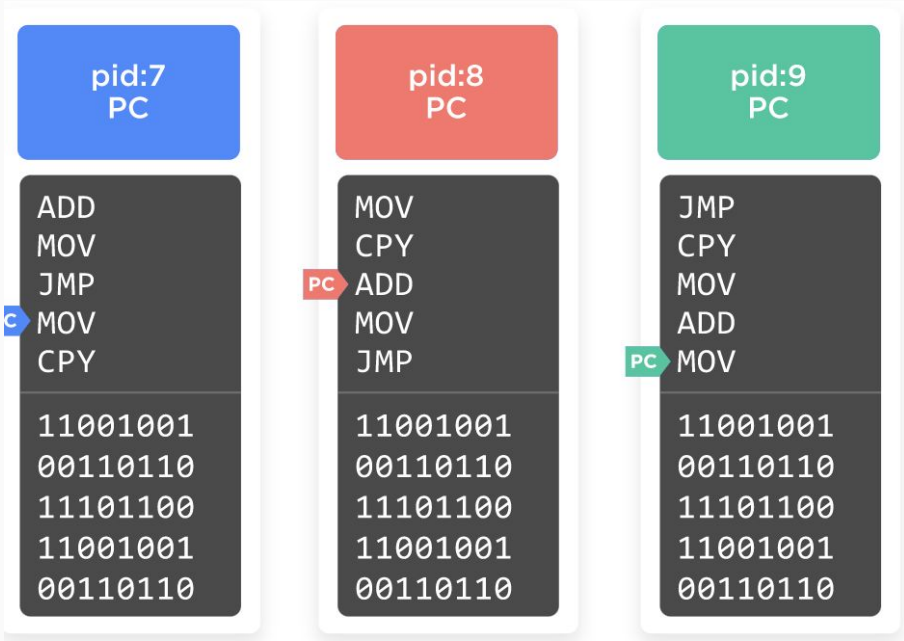
- Process hierarchy
- Each process has a unique ID
- Fork and exec process
- Init Process (PID 1)

OS Lies!

Process Asks OS:

- **To pass a message to another process**
  - Fake file!
- **Every Process can know about every other process**
  - What filesystems are available
  - What users are on the system
  - What permissions they have
  - What is the hostname
  - Network devices available
- **OS will give the same answer to all of processes**

Each process has its own contained memory

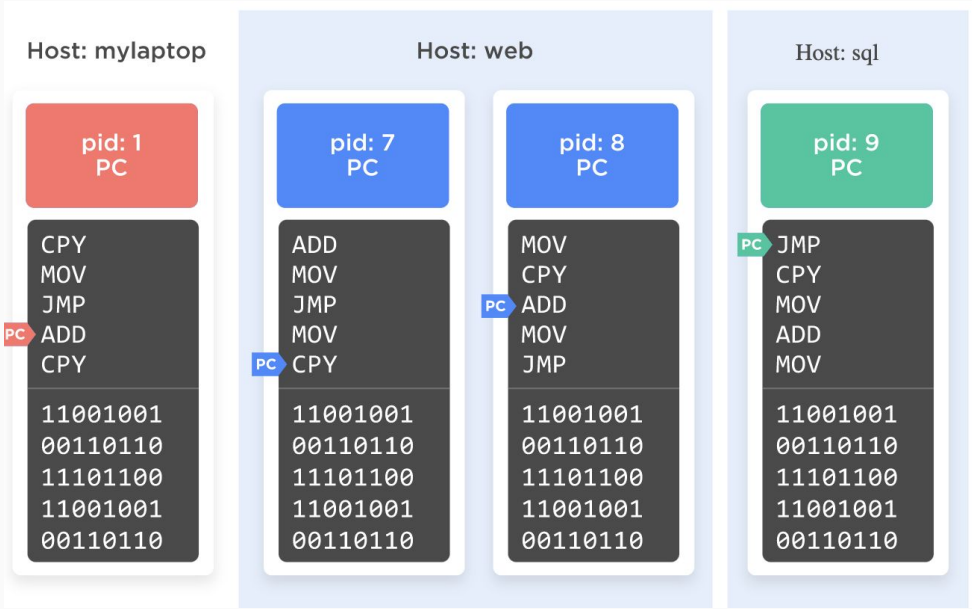


Linux Namespaces

- A way to lie to processes!
- Provide a way to segment groups of processes from each other
- Allow OS to lie to different sets of processes in different ways!
- Processes are in hierarchy!
  - Lying to parent means OS is lying to all of its children
- Namespaces are created using system calls
  - Programmatic way which process requests a service from OS

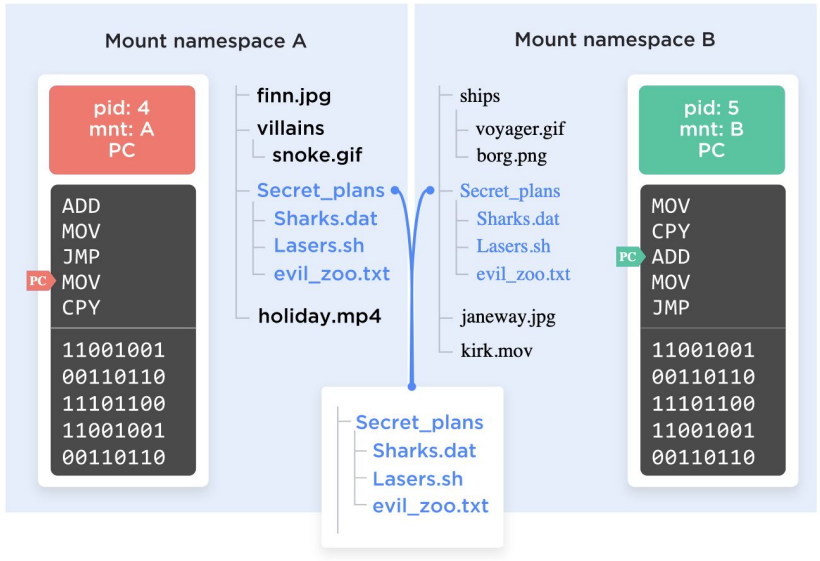
The UTS Namespace

- Controls the hostname of the computer
- `sethostname()` `setdomainname()`, and `uname()`



The mount Namespace

- Lets operating system present a different filesystem to a different set of processes
- `chroot()`
  - Lets a selected process to view a specific subset of filesystem as though it were the whole
  - Chroot jail

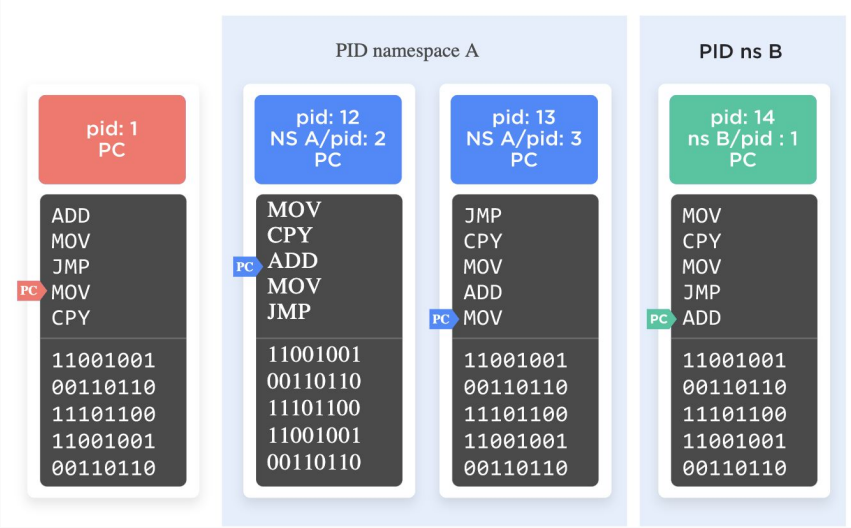


IPC Namespace

- IPC
  - The way processes talk to each other
- Makes communication possible only for processes inside a namespace

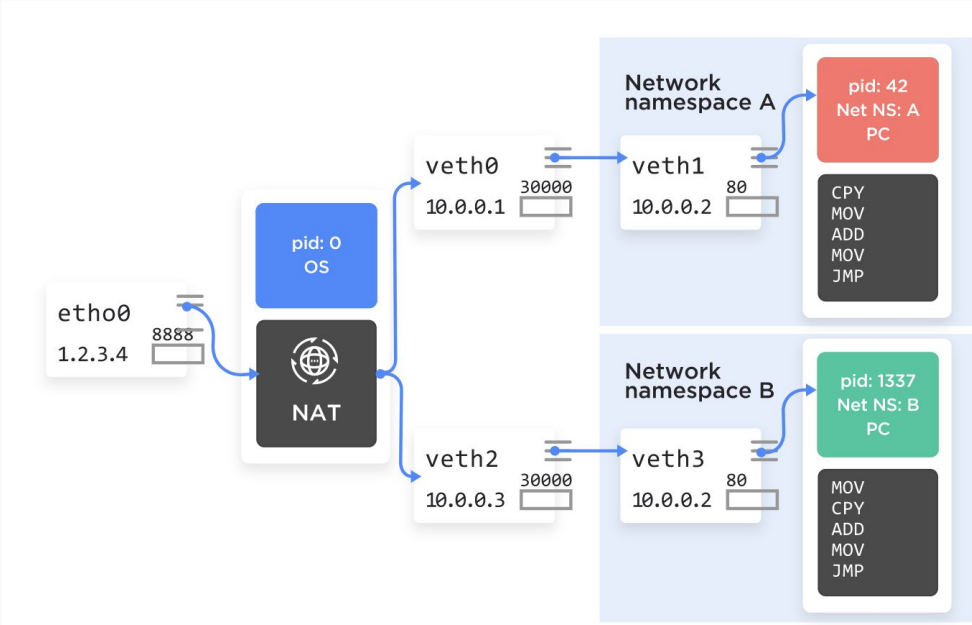
Process ID (PID) Namespace

- Remember process hierarchy ?
- You can see processes of all users!
- PID namespace abstracts PIDs in a namespace
- Process can’t initiate process if it doesn’t know they exist!



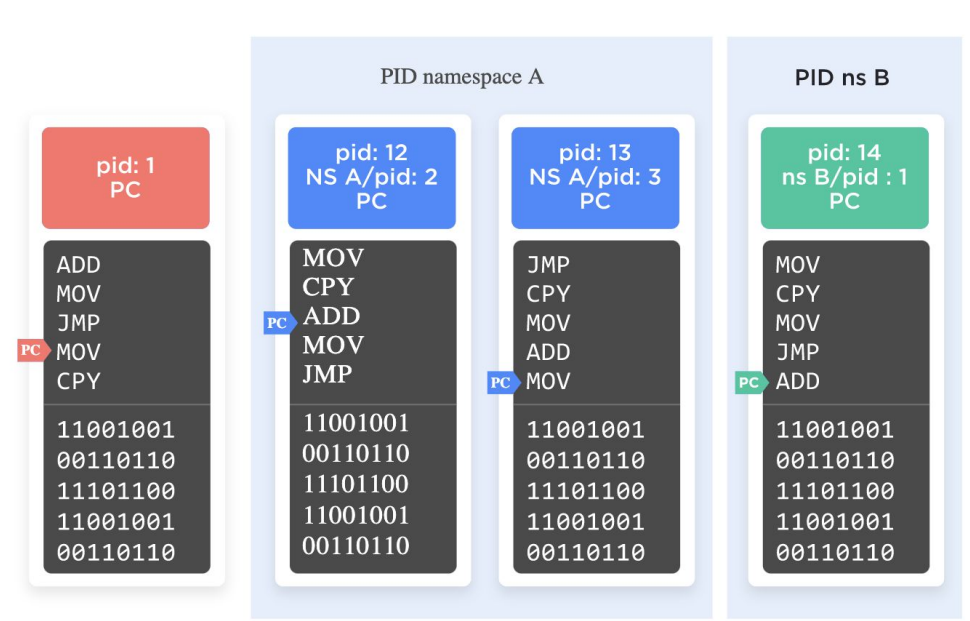
Network Namespace

- Allows creation of separate (virtual) network devices
- A network device can be used in only one device at a time
- Physical devices can only remain in root namespace



User Namespace

- Each process has a user and group
  - Manage access control
- Process is owned by any user (including root)



Control Groups

- Yet Another Lie!
- This time, about resources

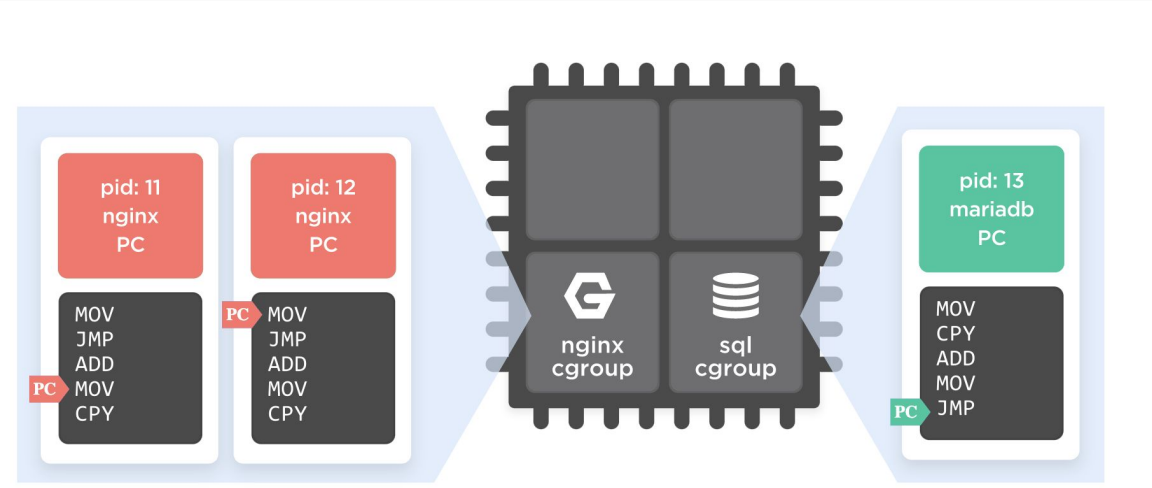
Control Groups

- Scheduler assigns CPU time to different processes
- How does it decide which processes are allowed to spend more time or less
- Computer has limited amount of memory
- How does it make sure one process doesn’t consume all of it ?

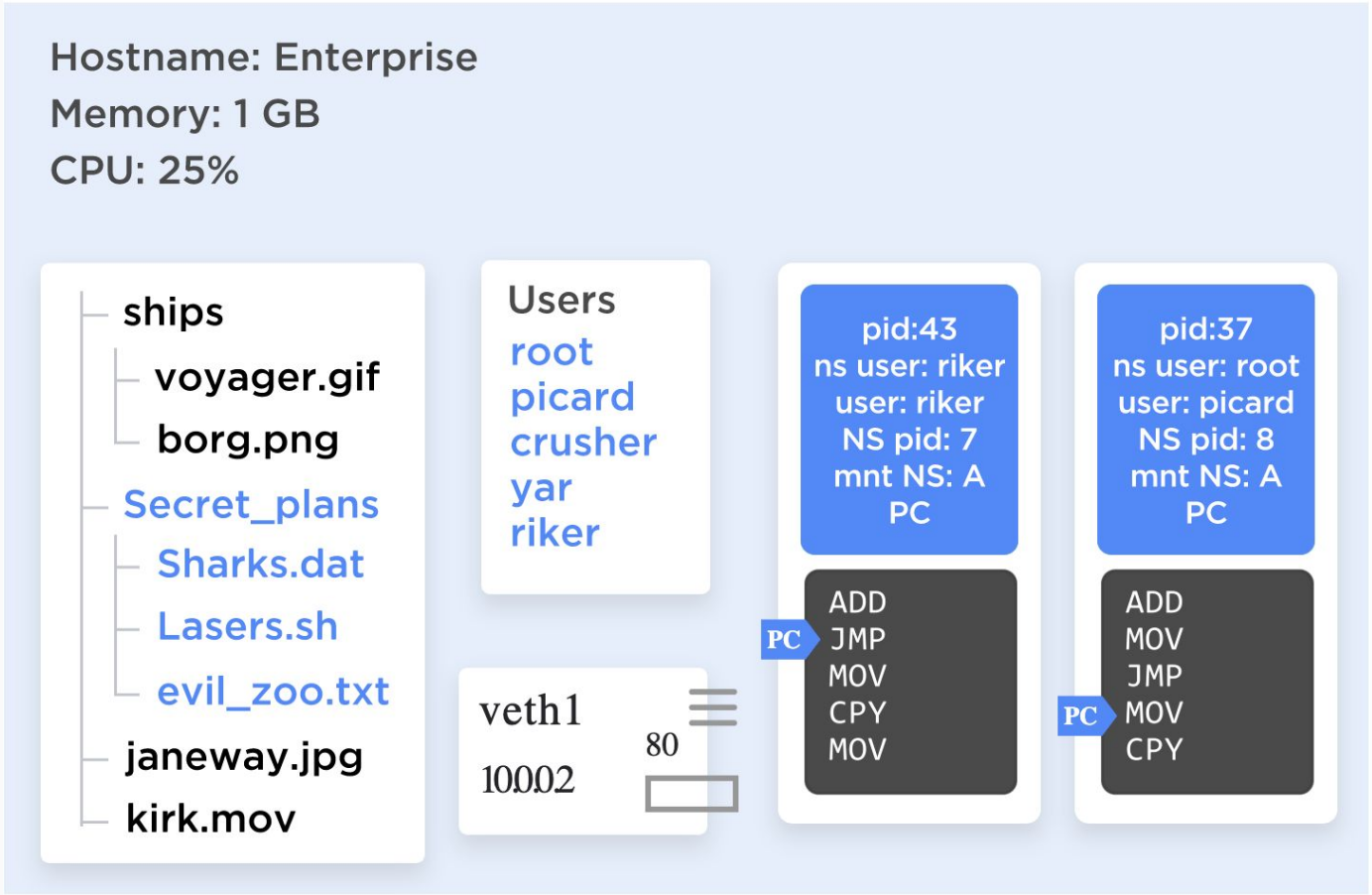
Control Groups!!

Control Groups

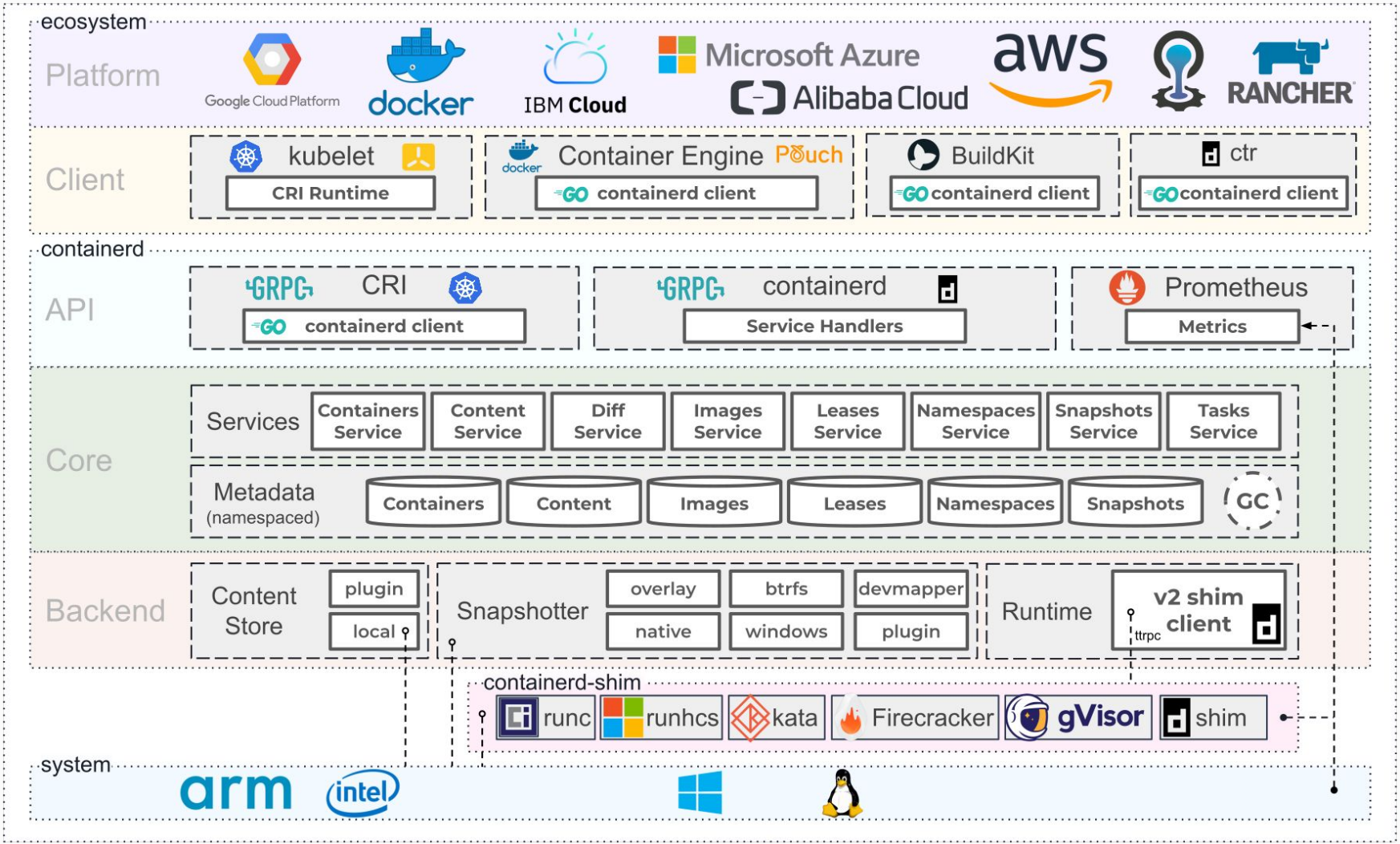
- Create a parallel hierarchy of processes
- Processes can be associated with one and only one leaf in that hierarchy
- Any node can have one or more **controllers** associated with it
  - Dozens of controllers
  - Some just track resource usage
  - Some limit
  - Some of them both
  - Most important: **CPU** and **Memory**



Lie, lie, lie !!!


Hostname: Enterprise
Memory: 1 GB
CPU: 25%
ships
voyager.gif
borg.png
Secret_plans
Sharks.dat
Lasers.sh
evil_zoo.txt
janeway.jpg
kirk.mov
Users
root
picard
crusher
yar
riker
veth1
10002
80
pid:43
ns user: riker
user: riker
NS pid: 7
mnt NS: A
PC
ADD
JMP
MOV
CPY
MOV
PC
pid:37
ns user: root
user: picard
NS pid: 8
mnt NS: A
PC
ADD
MOV
JMP
MOV
CPY
PC

Container Runtimes


ecosystem
Platform
Google Cloud Platform
docker
IBM Cloud
Microsoft Azure
Alibaba Cloud
aws
RANCHER
Client
kubelet
CRI Runtime
Container Engine
containerd client
BuildKit
containerd client
ctr
containerd client
containerd
API
GRPC
CRI
containerd client
GRPC
containerd
Service Handlers
Prometheus
Metrics
Core
Services
Containers Service
Content Service
Diff Service
Images Service
Leases Service
Namespaces Service
Snapshots Service
Tasks Service
Metadata
(namespaced)
Containers
Content
Images
Leases
Namespaces
Snapshots
GC
Backend
Content Store
plugin
local
Snapshotter
overlay
btrfs
devmapper
native
windows
plugin
Runtime
v2 shim client
ttrpc
containerd-shim
runc
runhcs
kata
Firecracker
gVisor
shim
system
arm
intel

Q&A

Sources

- https://blog.scottlowe.org/2013/09/04/introducing-linux-network-namespaces/
- https://www.youtube.com/watch?v=8fi7uSYlOdc&vl=en
- https://platform.sh/blog/2020/the-container-is-a-lie/
-