

Long module with lots of waffling



Architecture and Theory



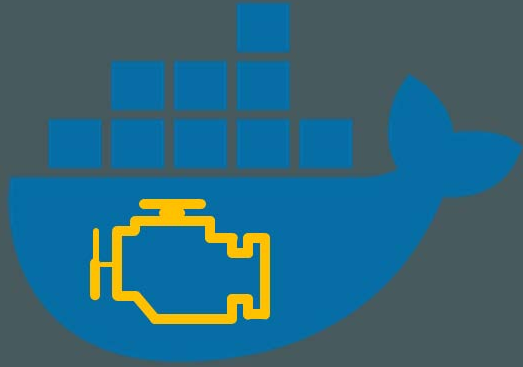
Nigel Poulton

@nigelpoulton www.nigelpoulton.com





Module Outline



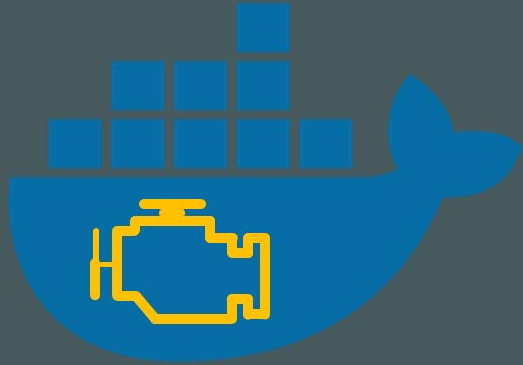
Waffle

Waffle

Waffle



Module Outline



Big picture view

Kernel primitives

Docker Engine

Windows spotlight

Recap





Domain 3: Installation and Configuration

- Demonstrate the ability to upgrade the Docker engine
- Understand namespaces and cgroups

Coming up

The Big Picture

The Big Picture

Kernel stuff & the Docker Engine



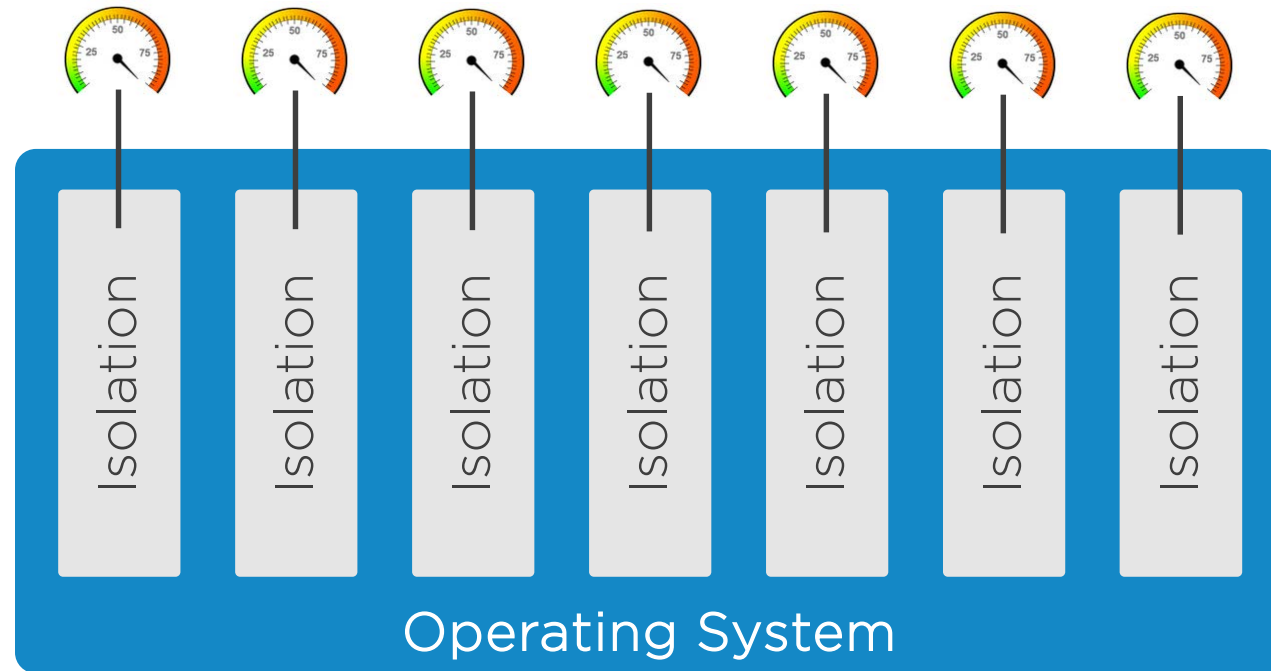
The Big Picture

Kernel stuff & the Docker Engine

container

/kən'teɪnə/

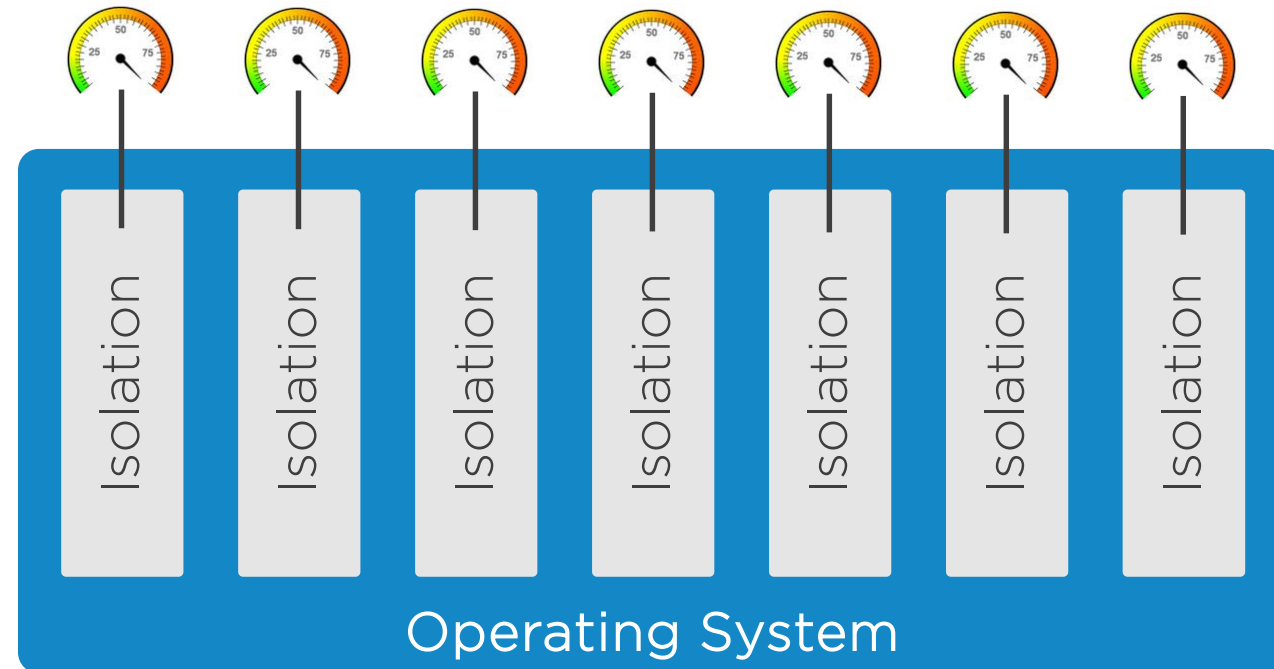
noun: Isolated area of an OS with resource usage limits applied



container

/kən'teɪnə/

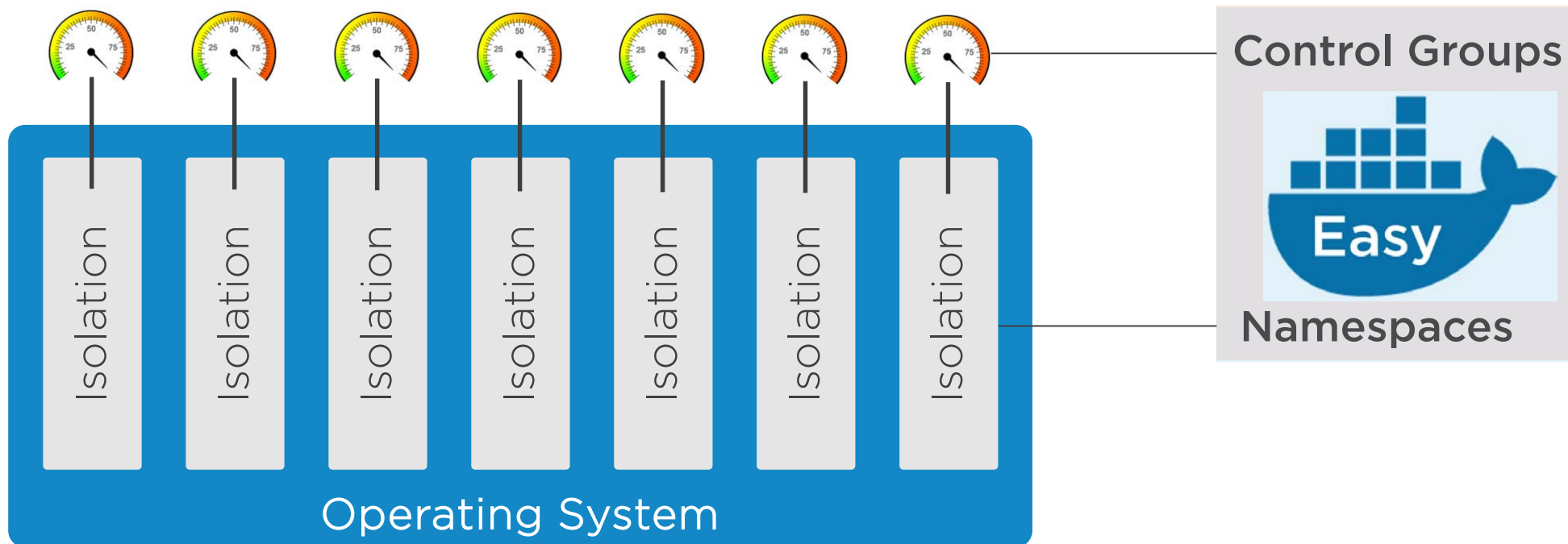
noun: Isolated area of an OS with resource usage limits applied



container

/kən'teɪnə/

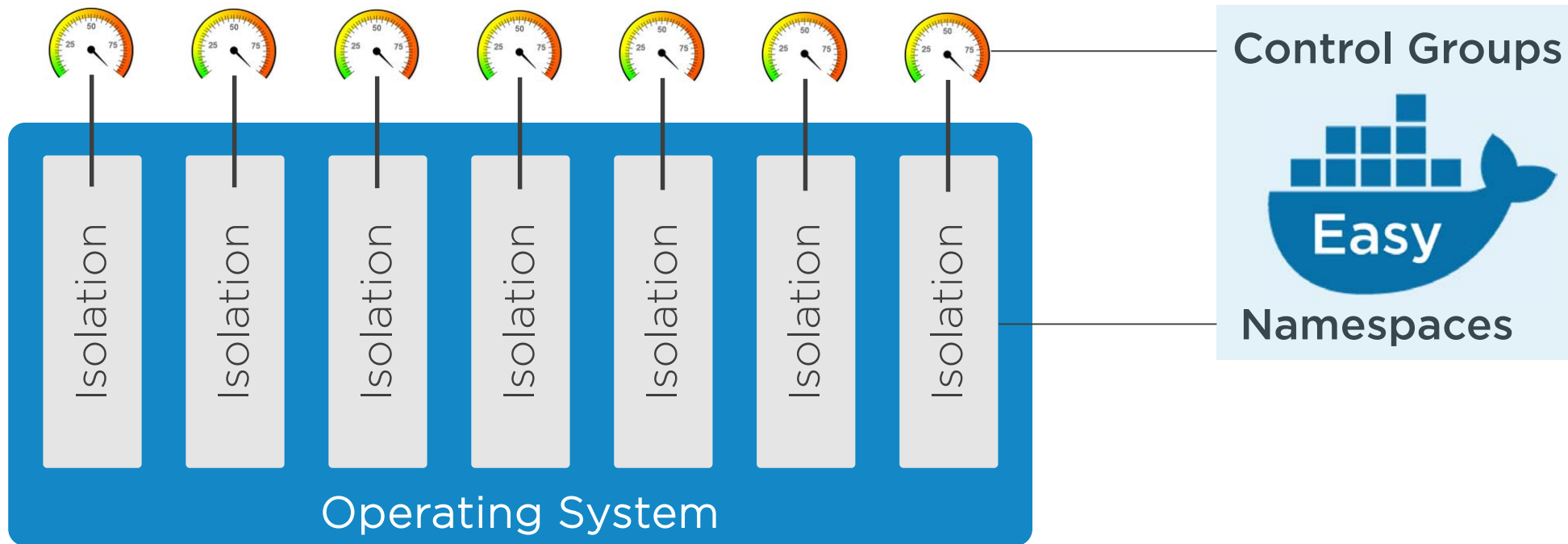
noun: Isolated area of an OS with resource usage limits applied



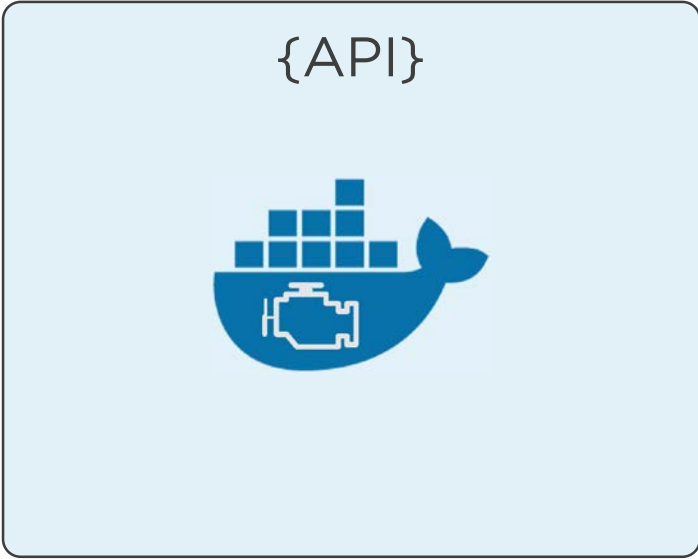
container

/kən'teɪnə/

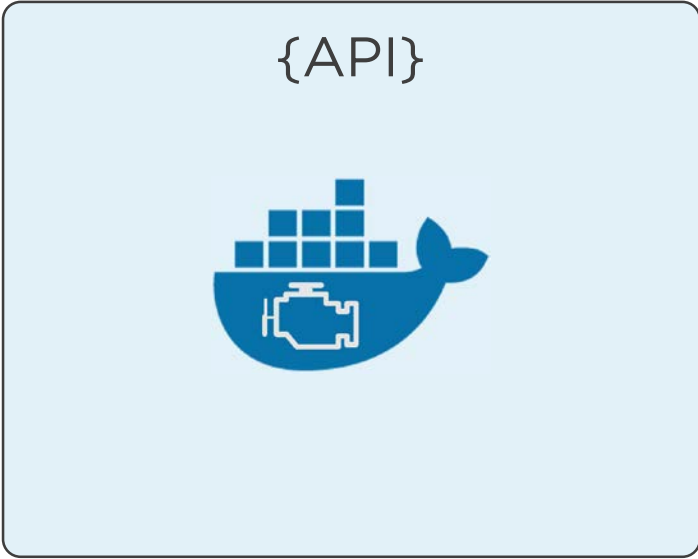
noun: Isolated area of an OS with resource usage limits applied

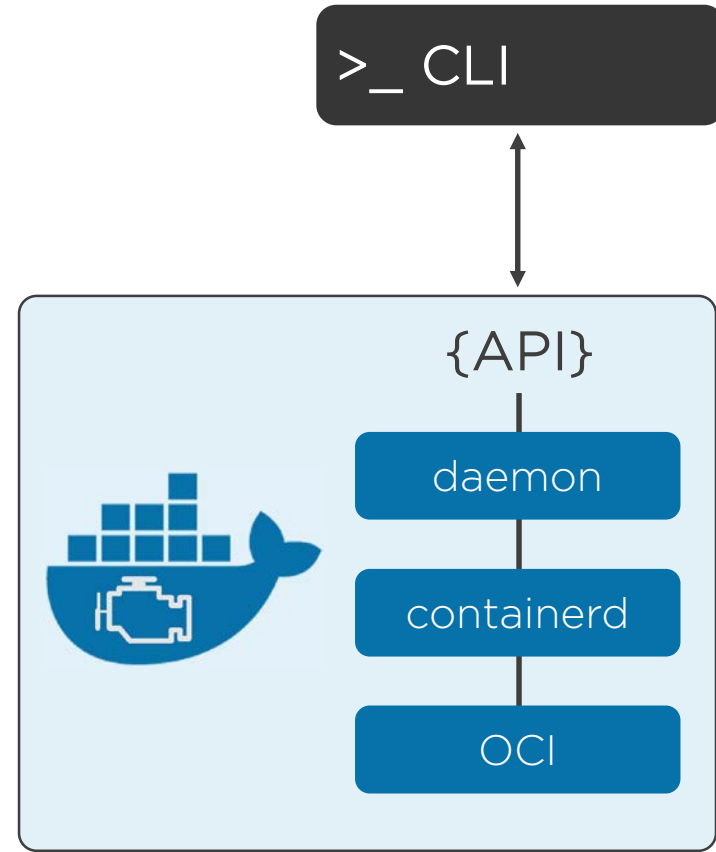


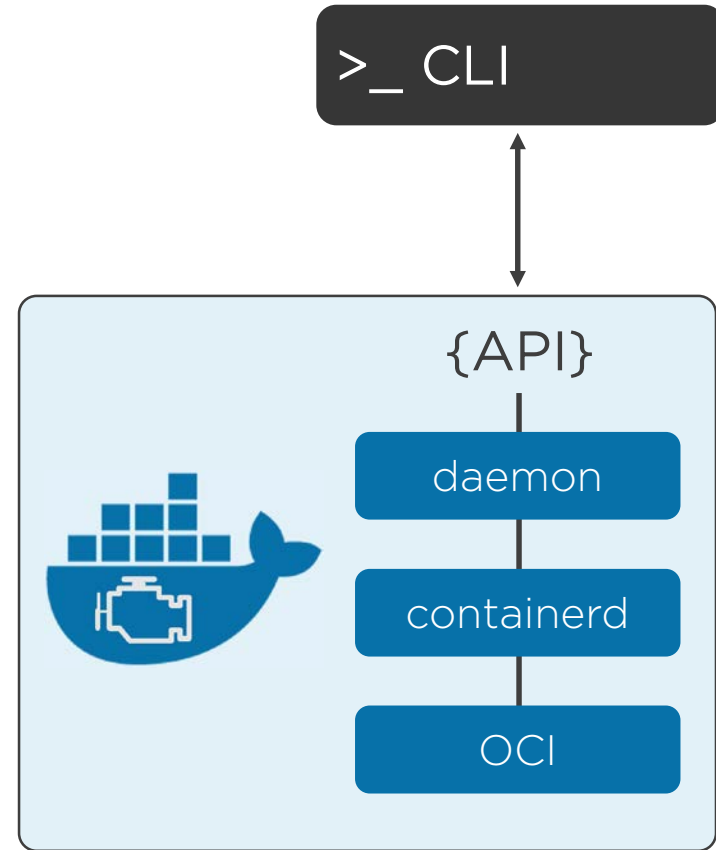
>_ CLI

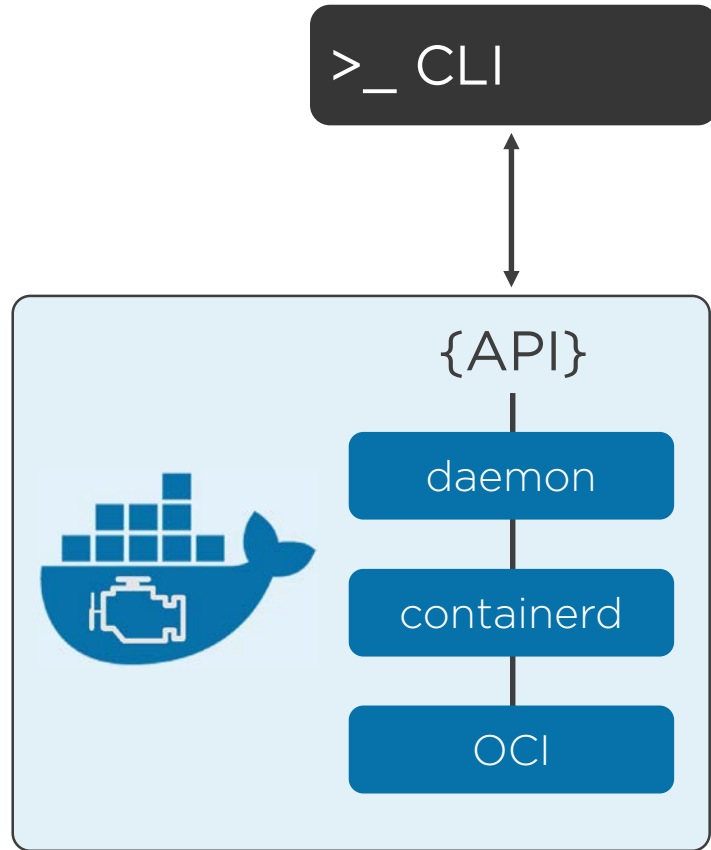


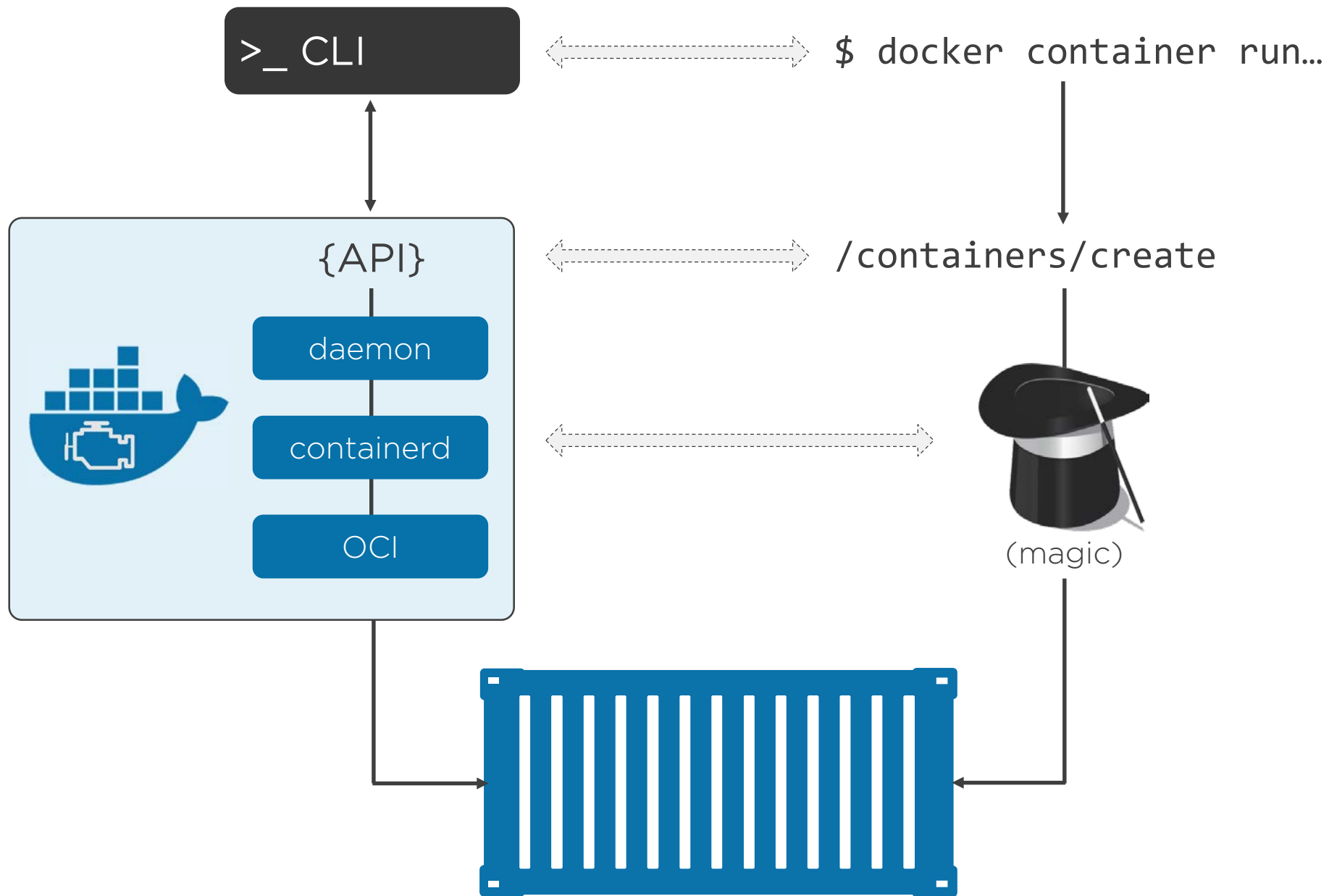
>_ CLI









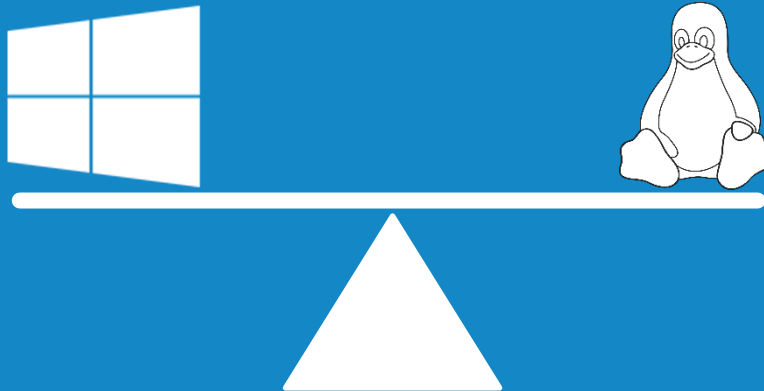


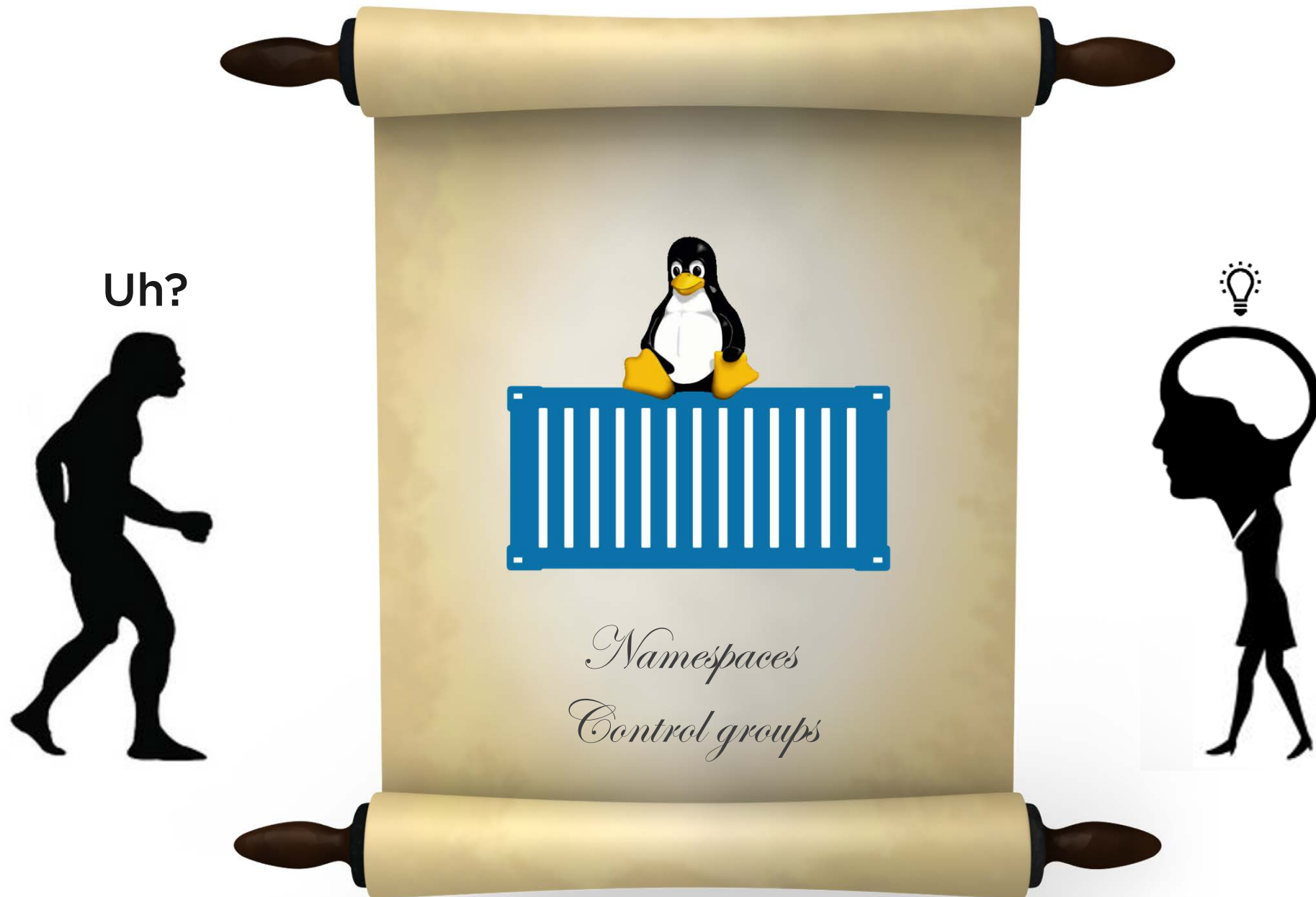
Coming up Kernel Internals

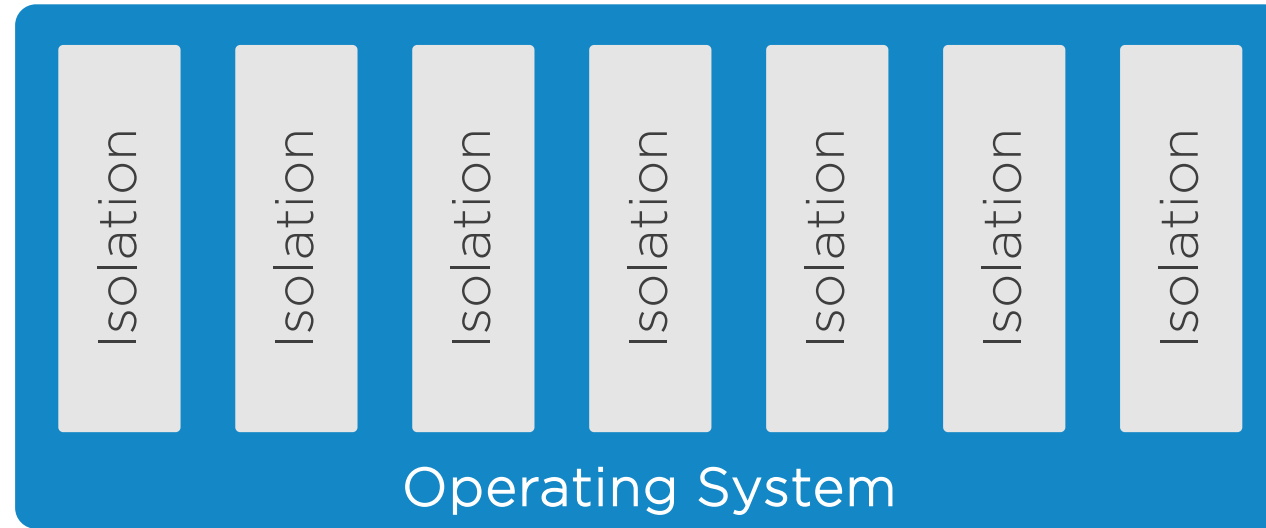


Kernel Internals

Container primitives in the kernel





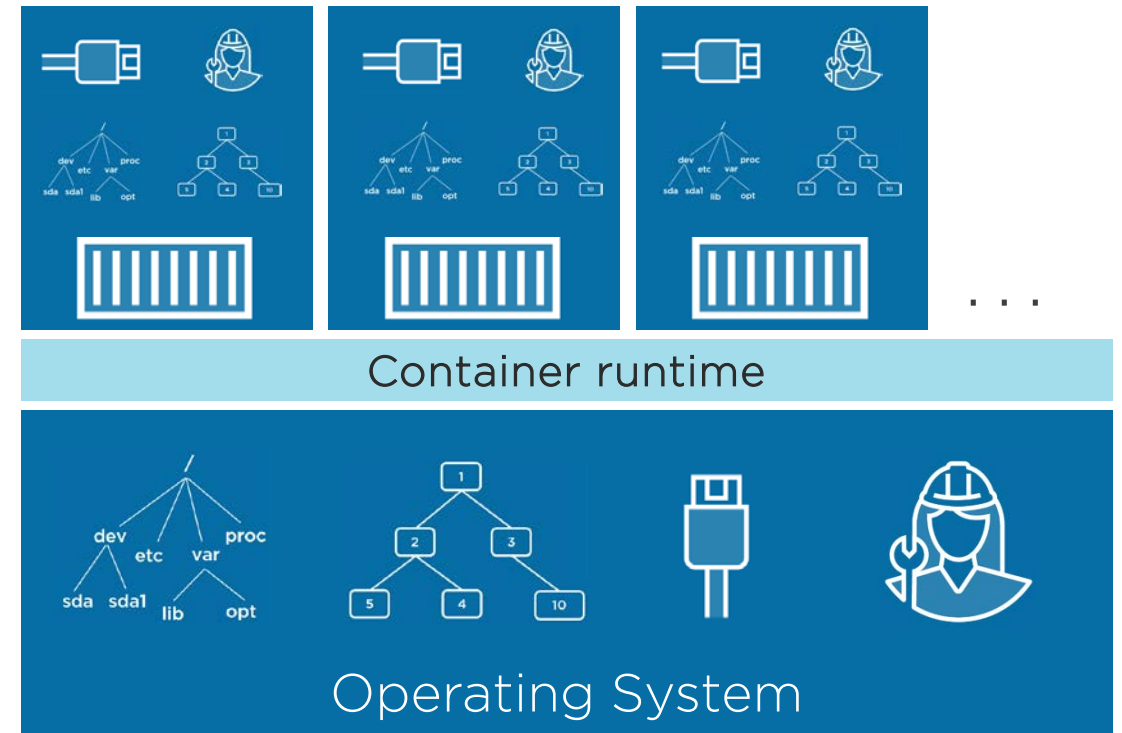
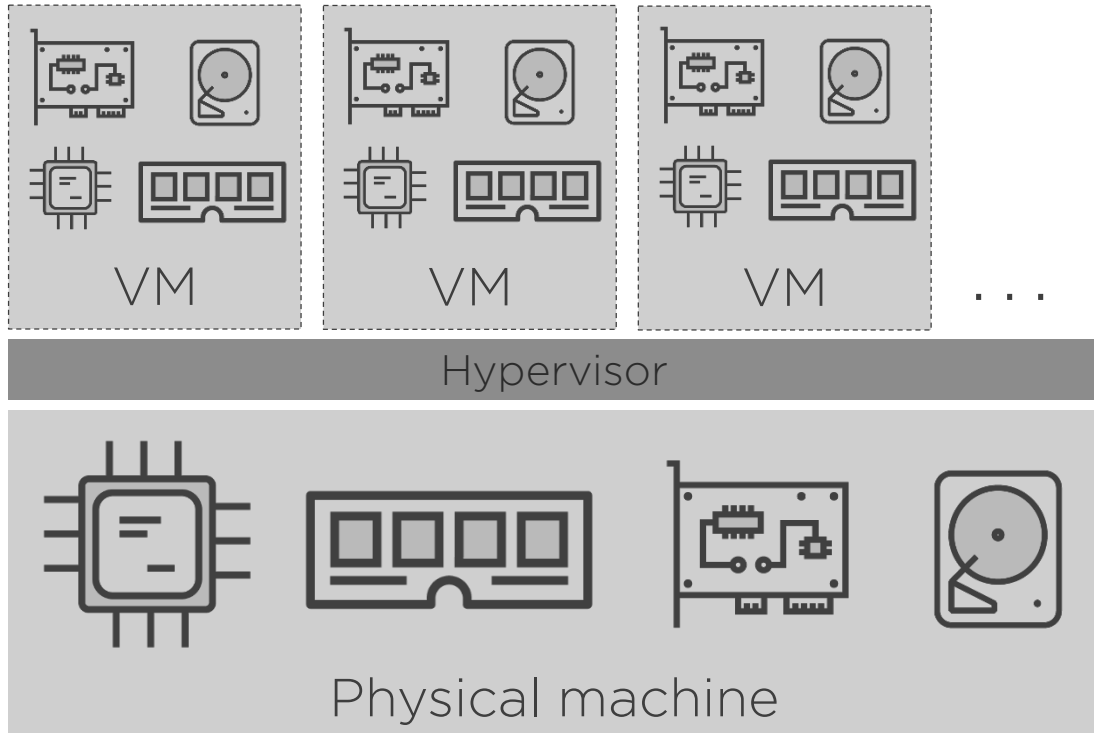


Namespaces



Control Groups



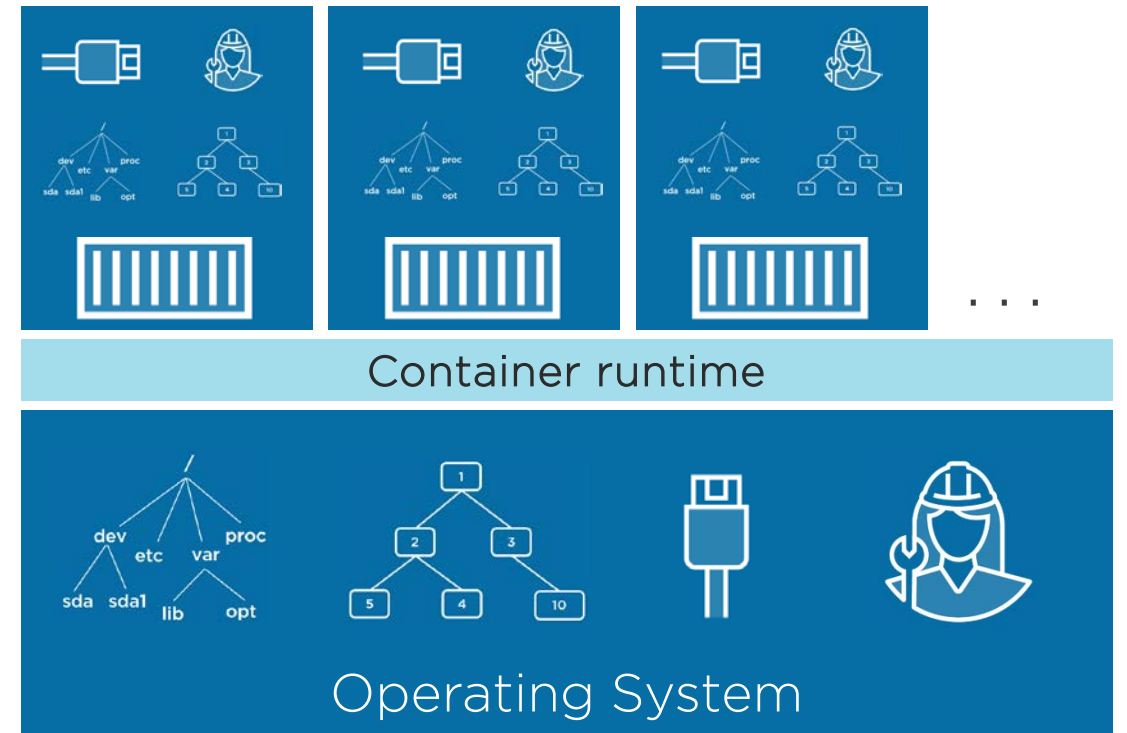
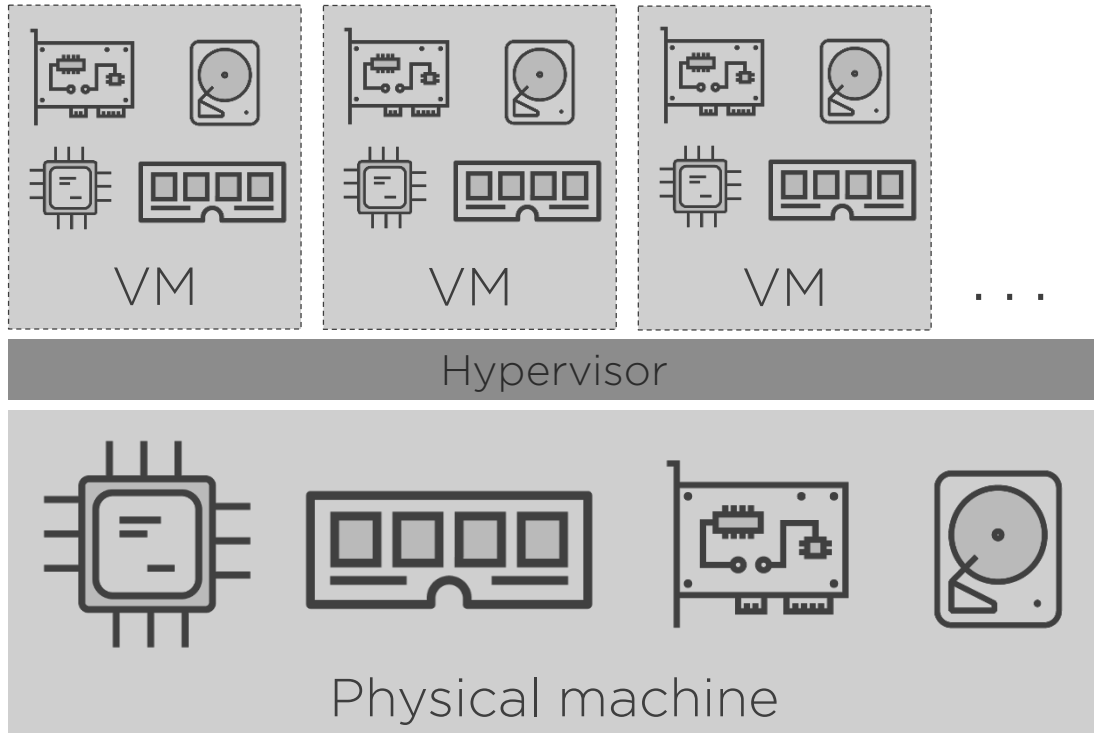


Namespaces



Control Groups



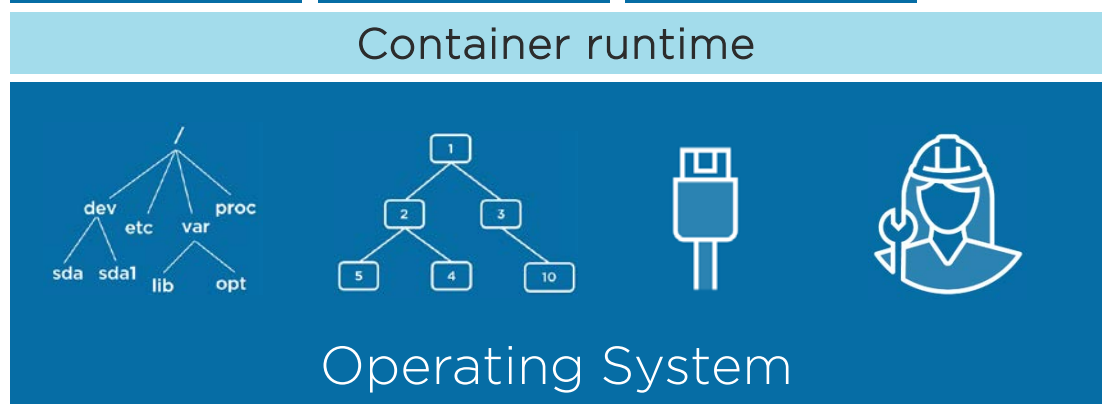
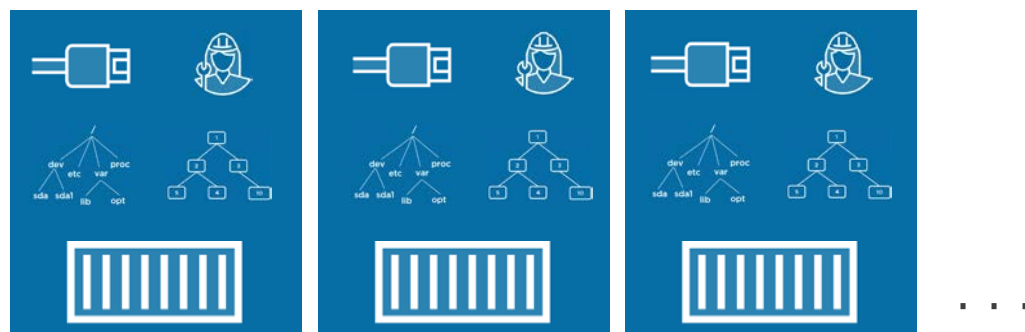


Namespaces



Control Groups



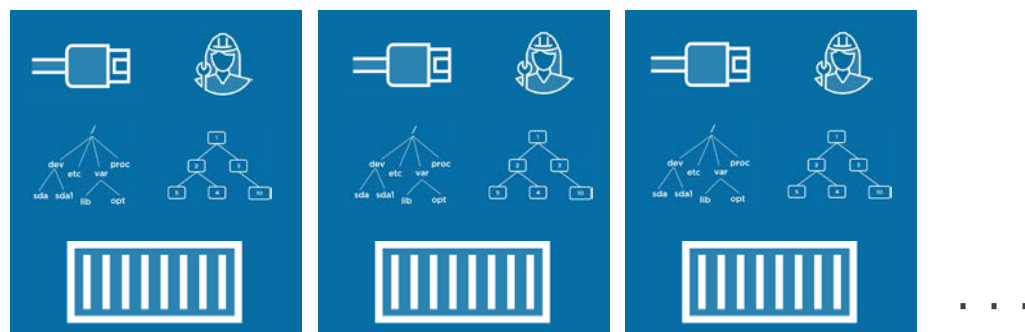


Namespaces

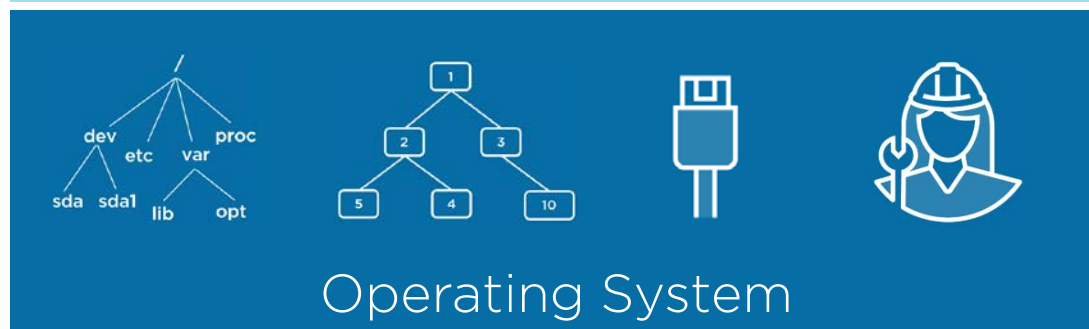


Control Groups





Container runtime



Operating System

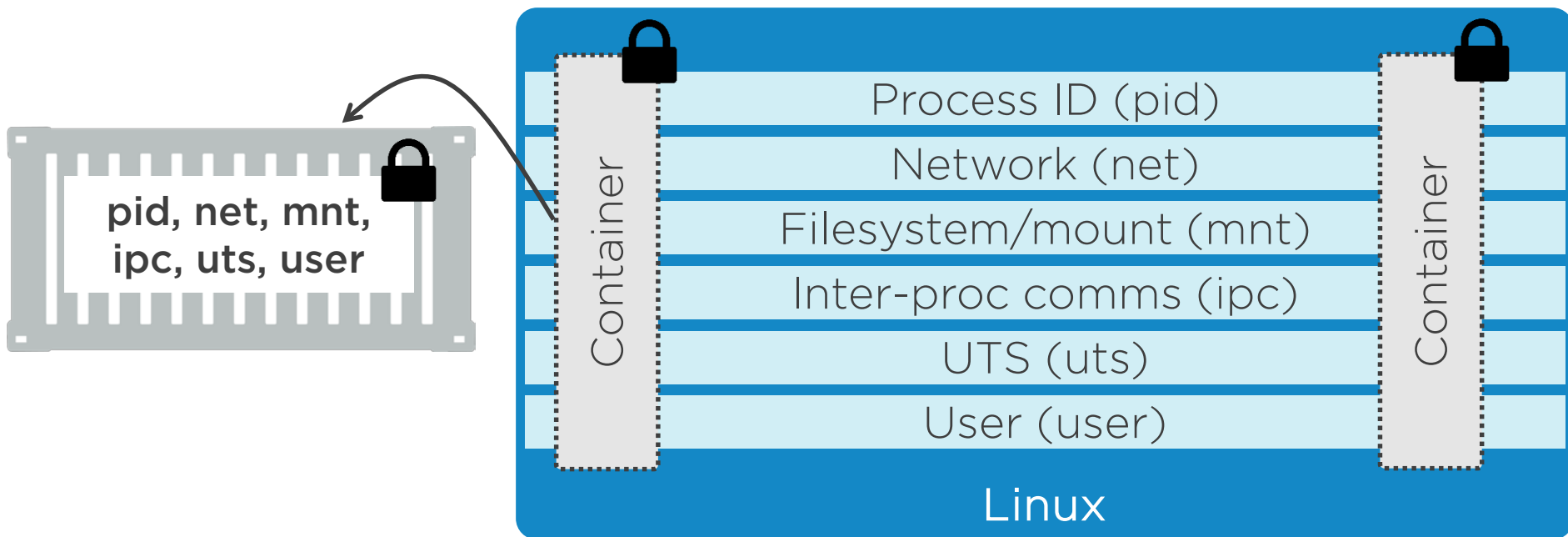


Namespaces



Control Groups



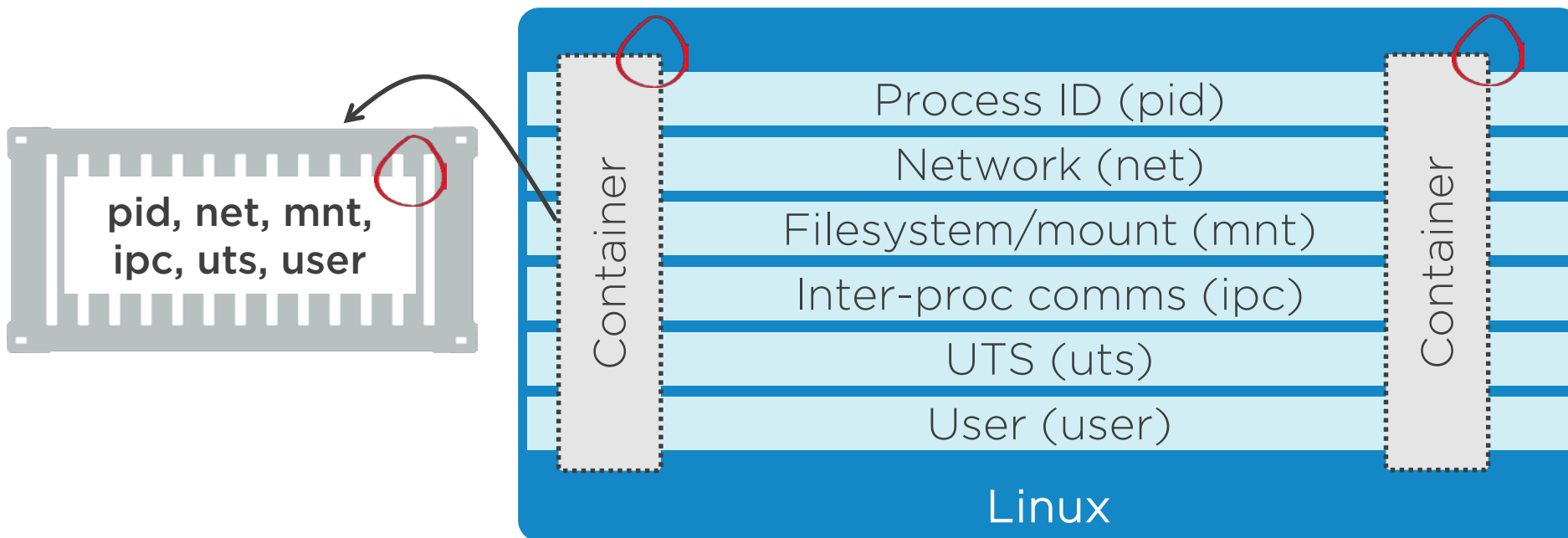


Namespaces



Control Groups



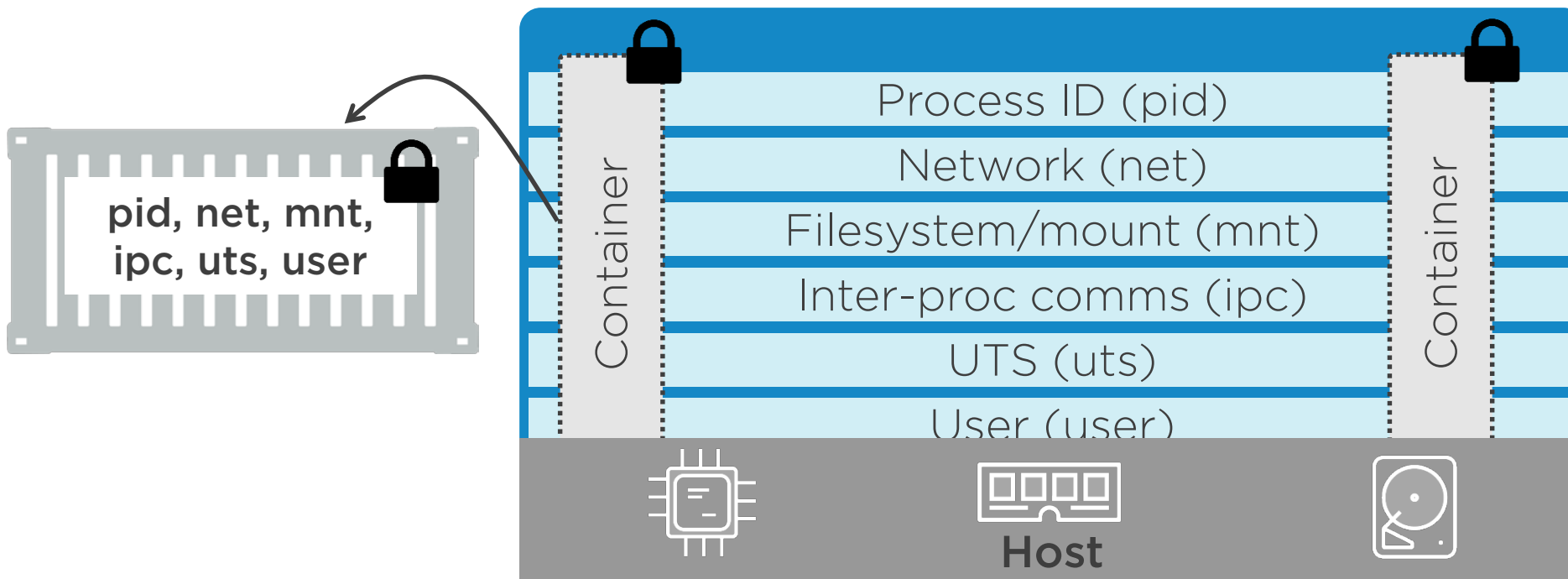


Namespaces



Control Groups





Namespaces

Linux

- pid
- net
- mount
- ...

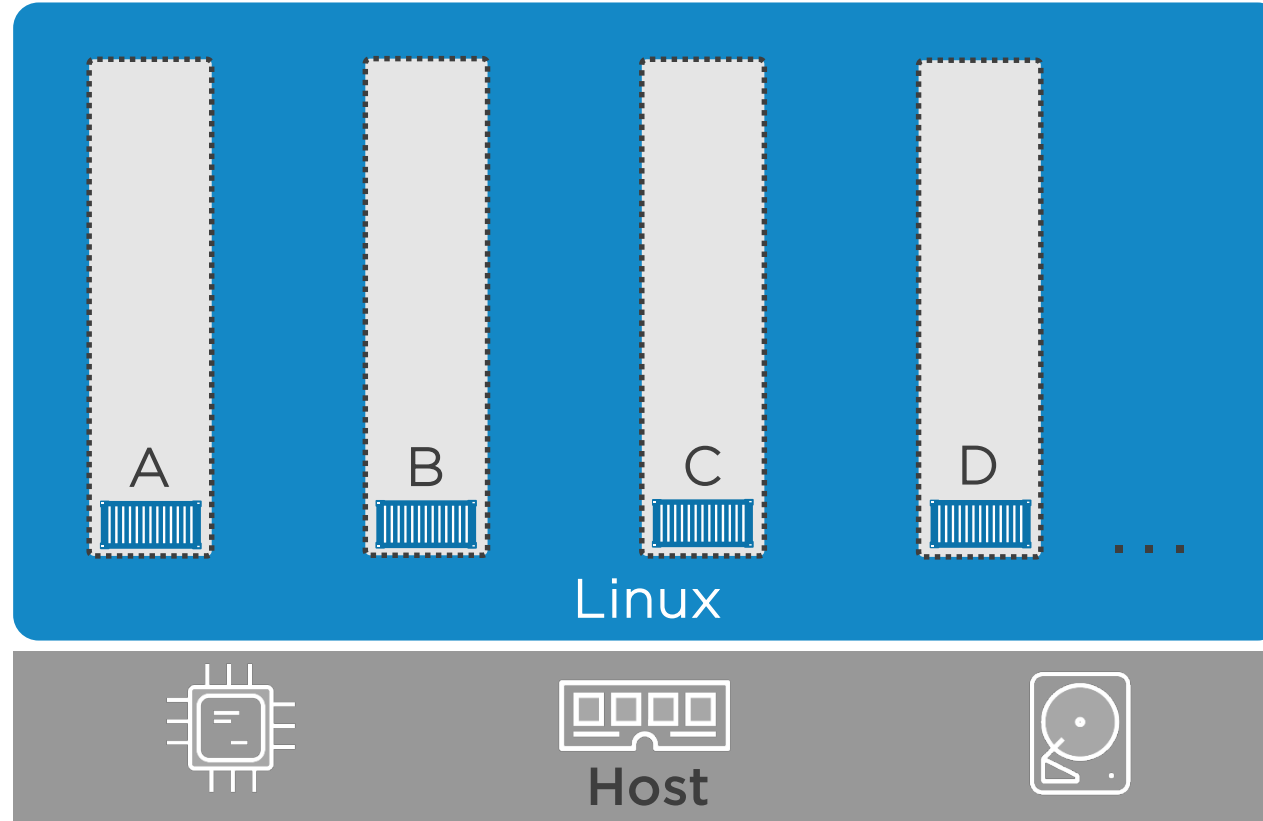
Windows

- object
- proc table
- networking
- ...



Control Groups





Namespaces

Linux

- pid
- net
- mount
- ...

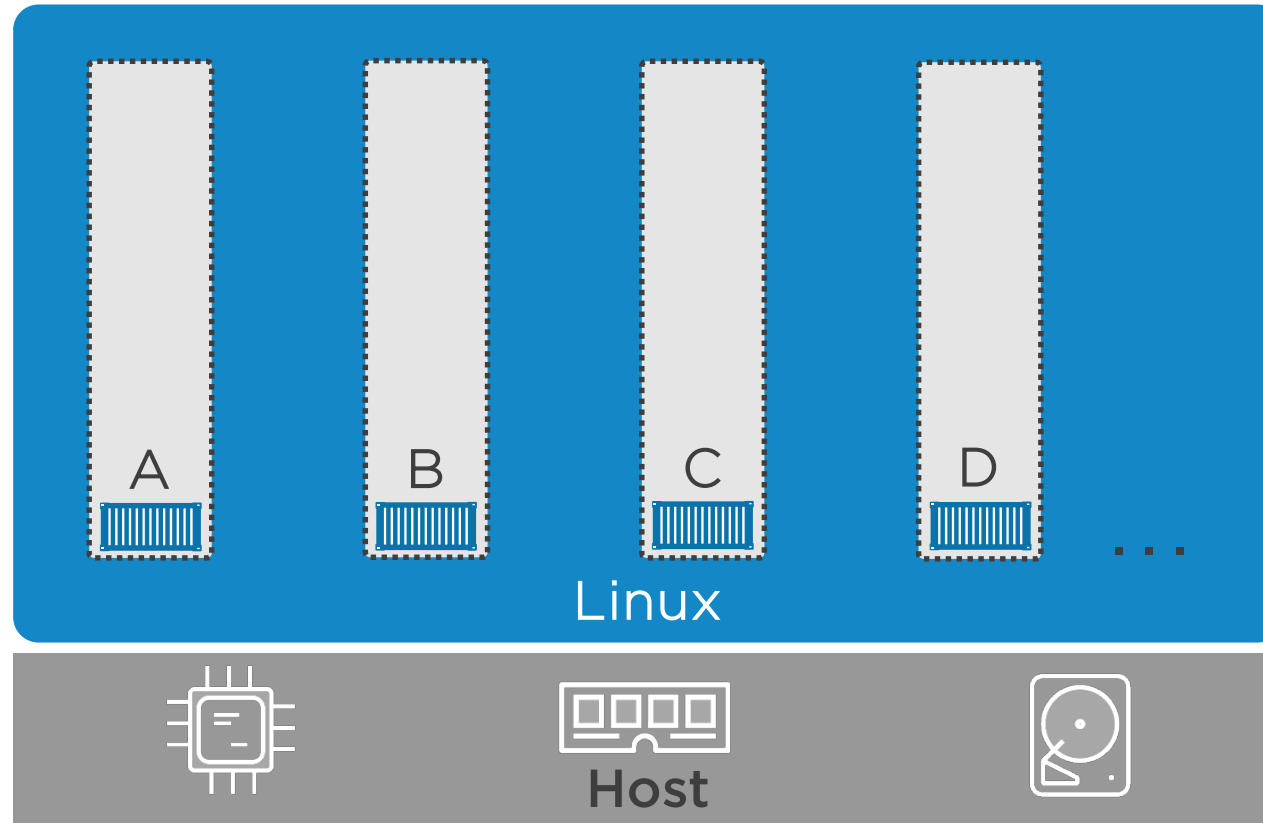
Windows

- object
- proc table
- networking
- ...



Control Groups





Namespaces

Linux

- pid
- net
- mount
- ...

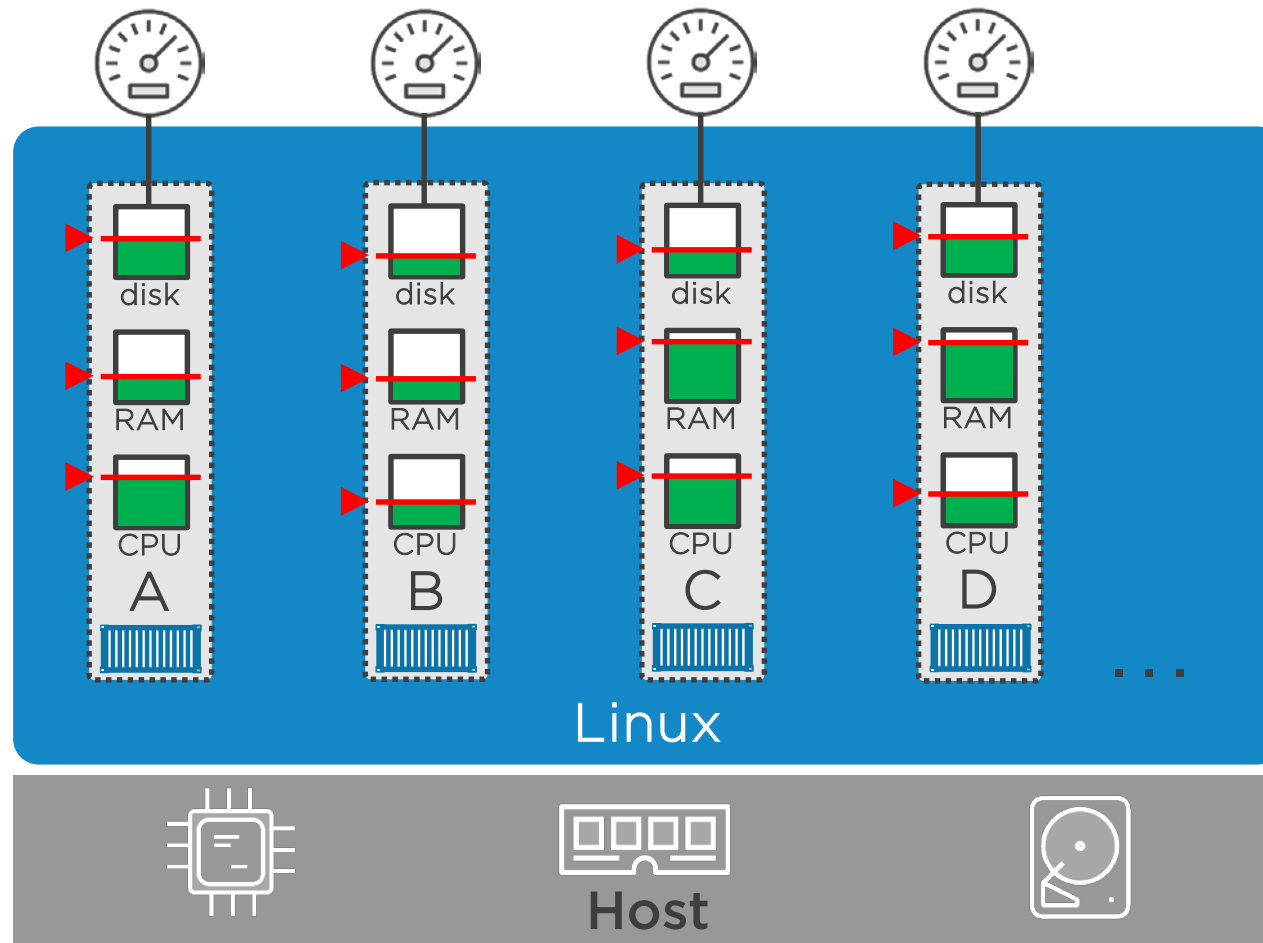
Windows

- object
- proc table
- networking
- ...



Control Groups (Windows a.k.a. Job Objects)





Namespaces

Linux

- pid
- net
- mount

...

Windows

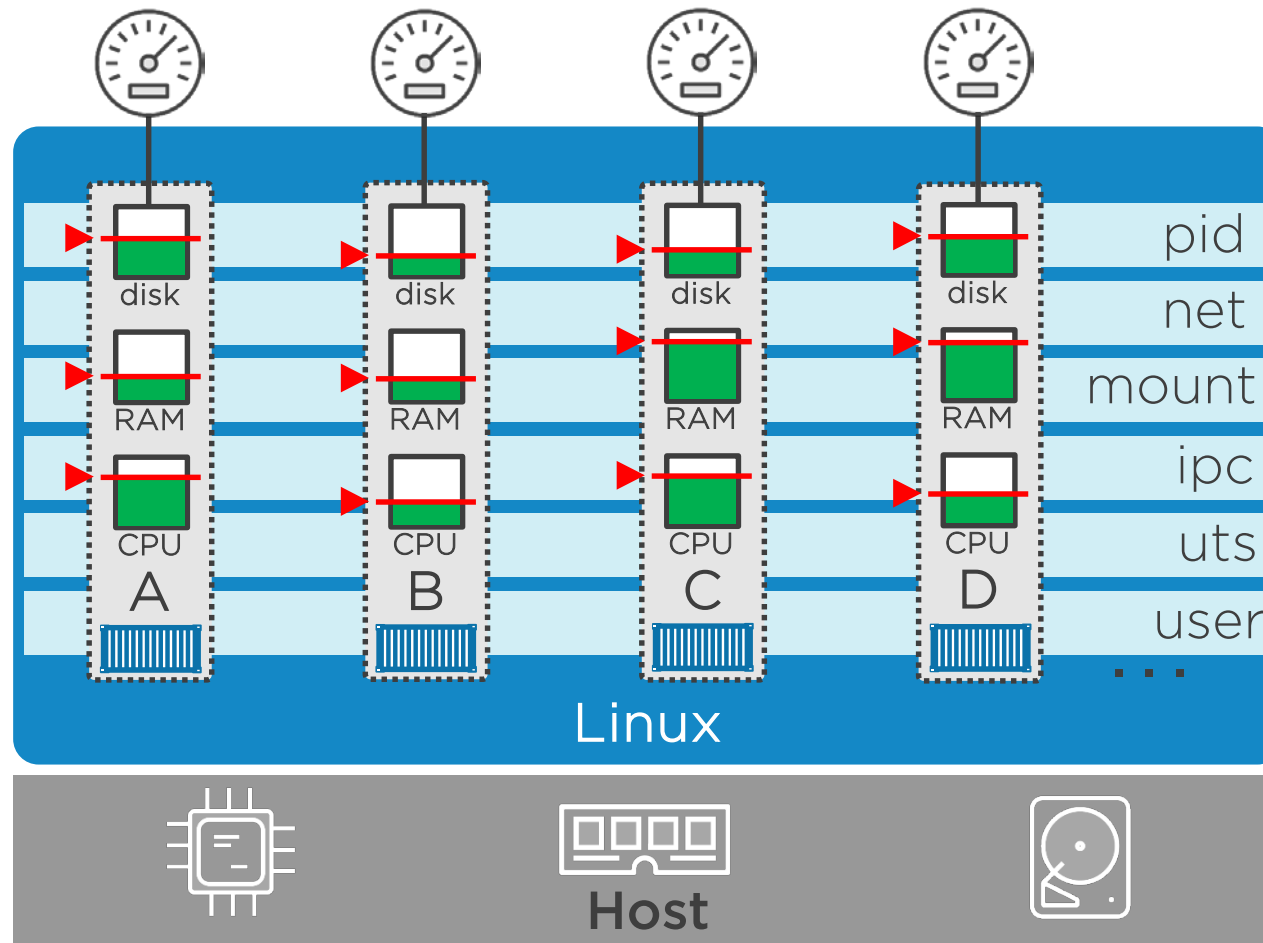
- object
- proc table
- networking

...



Control Groups (Windows a.k.a. Job Objects)





Layers



Union fs/mount
& CoW

Namespaces

Linux

- pid
- net
- mount
- ...

Windows

- object
- proc table
- networking
- ...

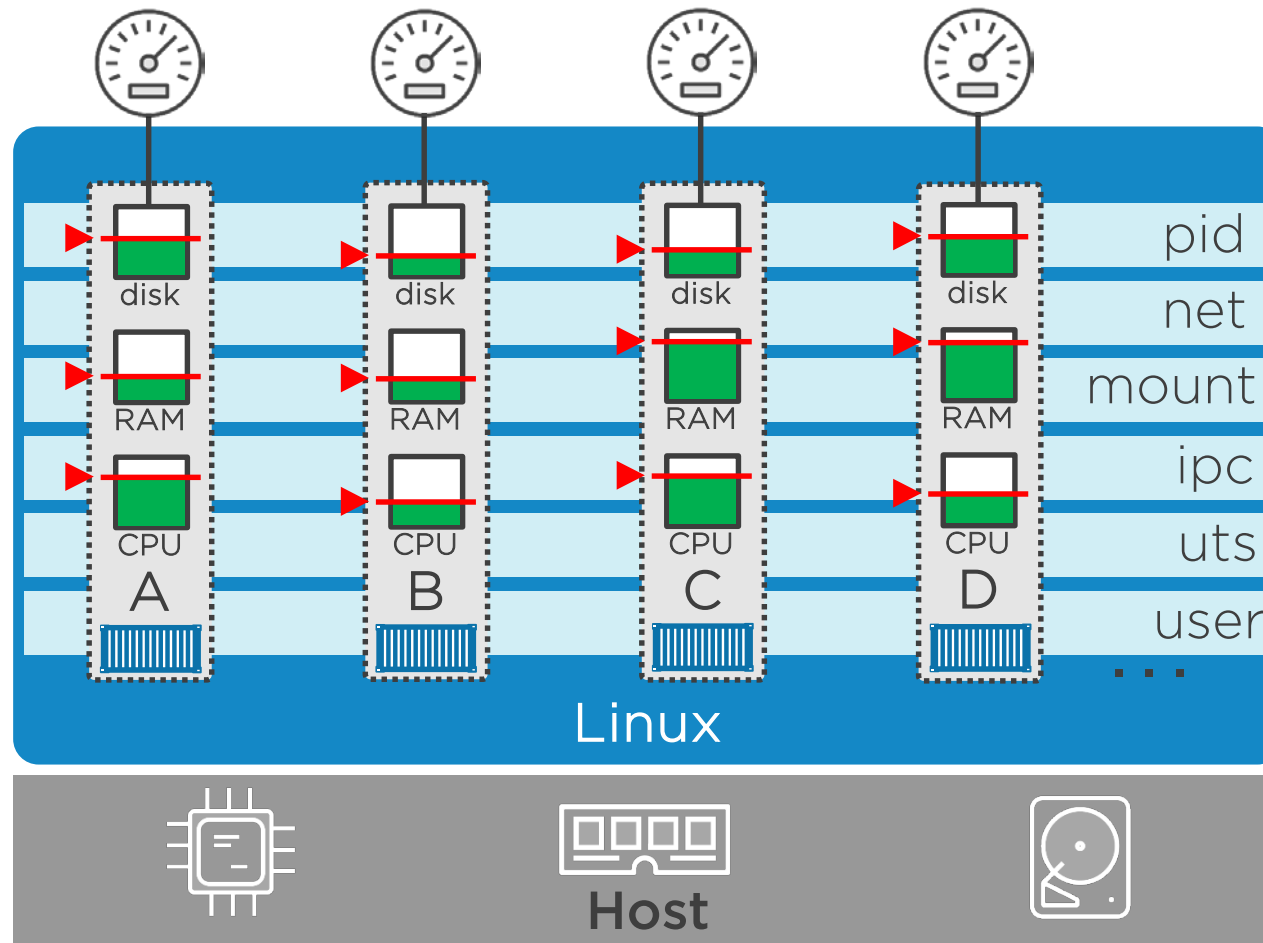


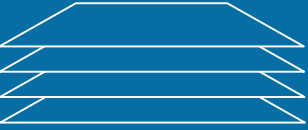


Control Groups

(Windows a.k.a. Job Objects)

- Grouping processes
- Imposing resource limits





Layers	Namespaces	Control Groups
 <p>Union fs/mount & CoW</p>	<div> <div>  <p>Linux</p> <ul style="list-style-type: none"> - pid - net - mount ... </div> <div> <p>Windows</p> <ul style="list-style-type: none"> - object - proc table - networking ... </div> </div>	<div>  <p>Control Groups (Windows a.k.a. Job Objects)</p> <ul style="list-style-type: none"> - Grouping processes - Imposing resource limits </div>

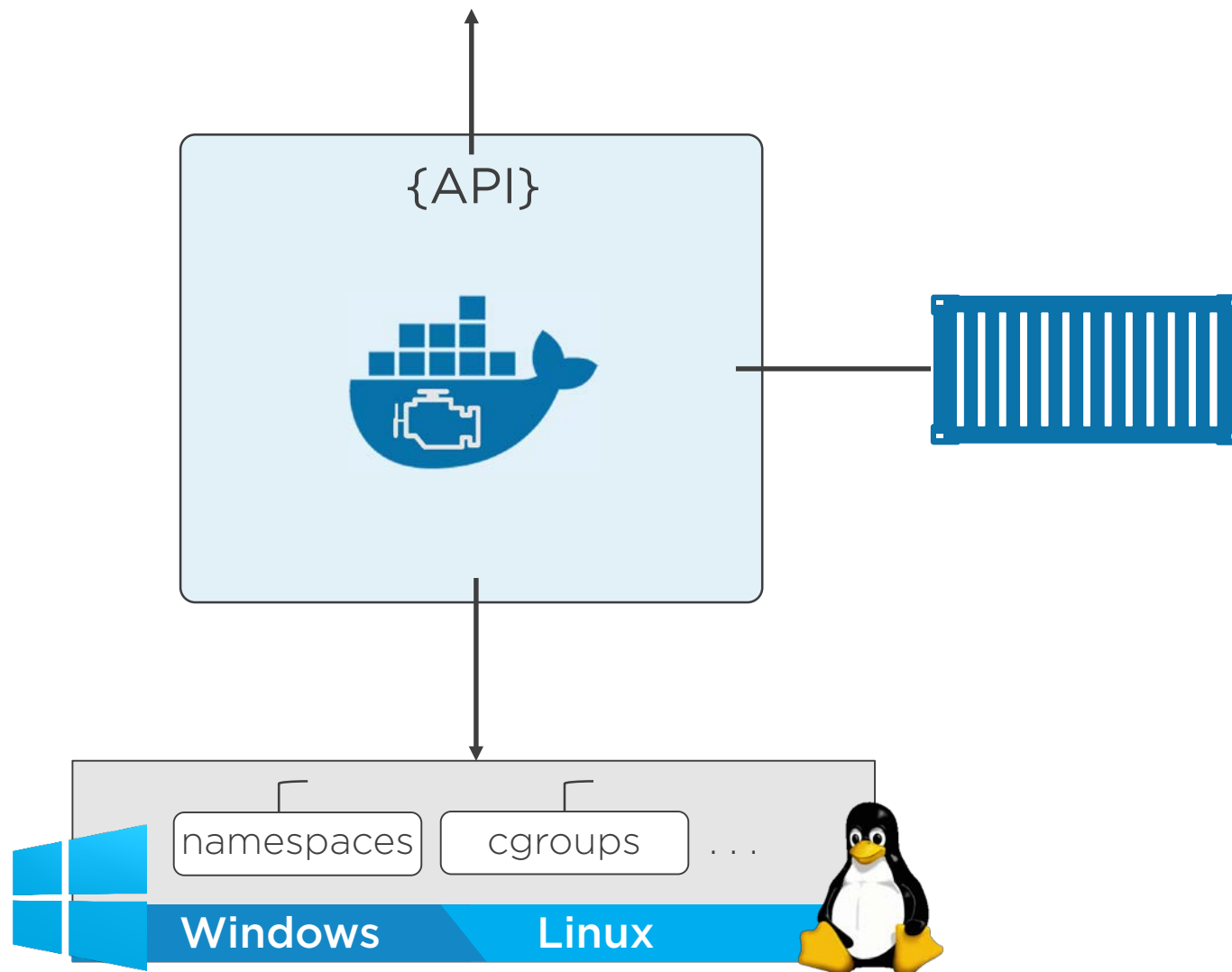
Coming up
The Docker Engine

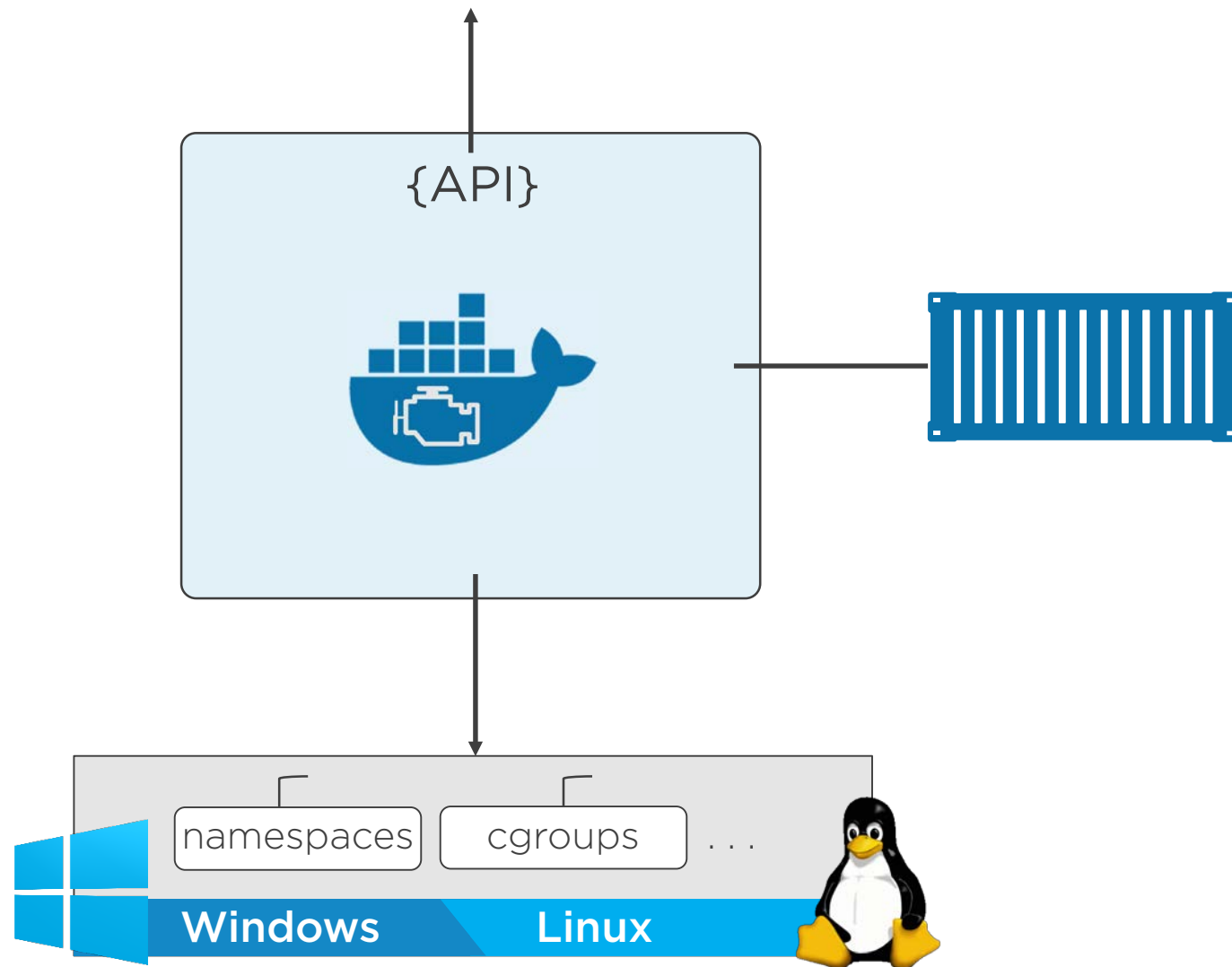


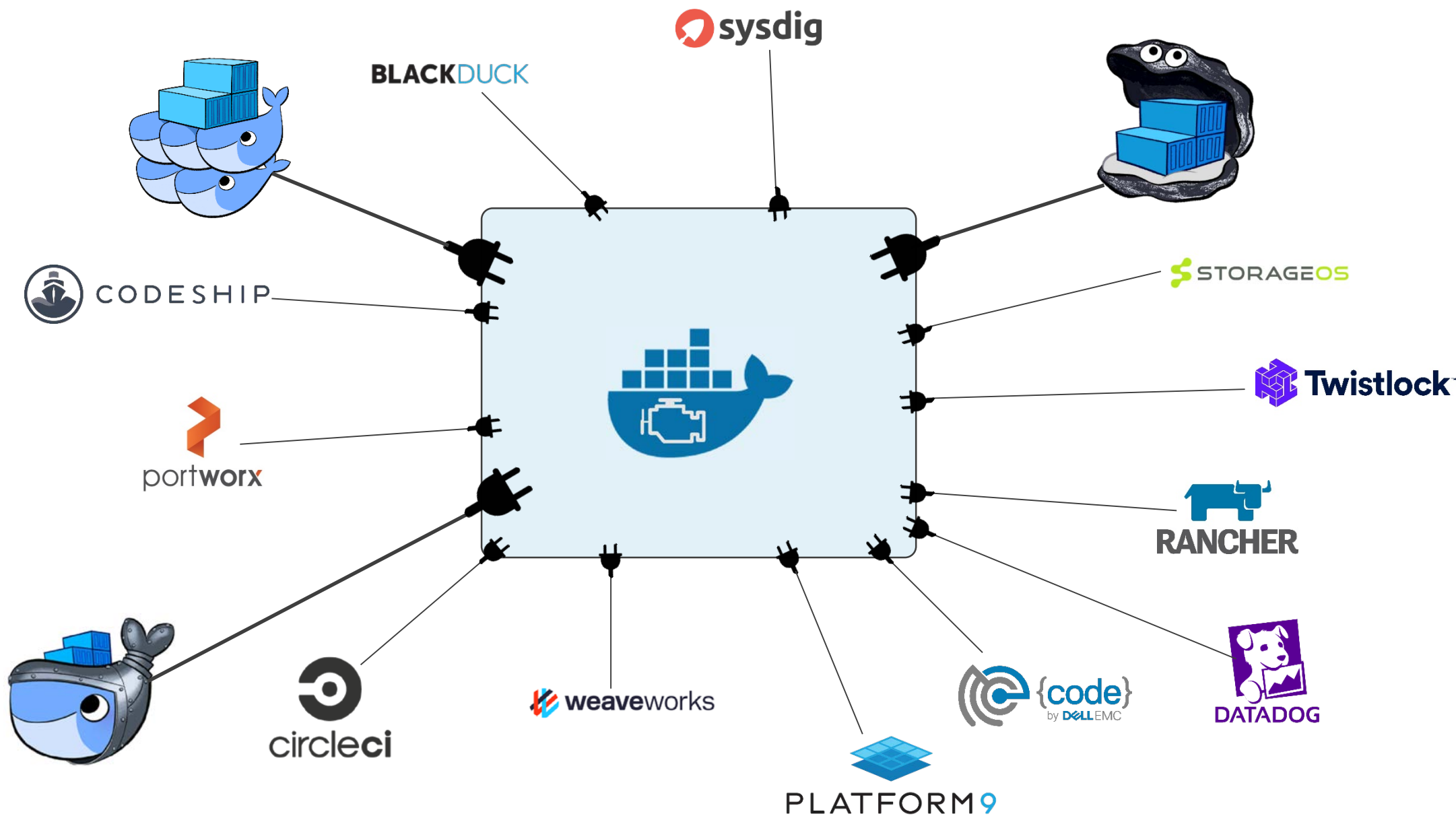
The Docker Engine

Making containers easy





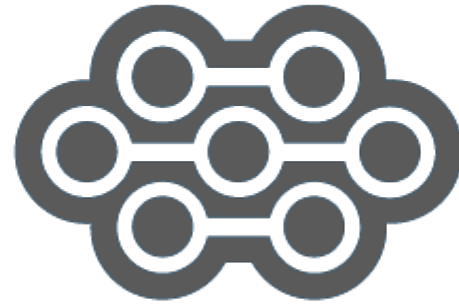




*Random logos from DockerCon sponsor list







dotCloud



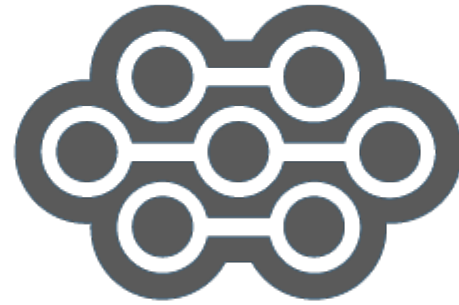
LXC

dc



AUFS

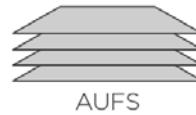


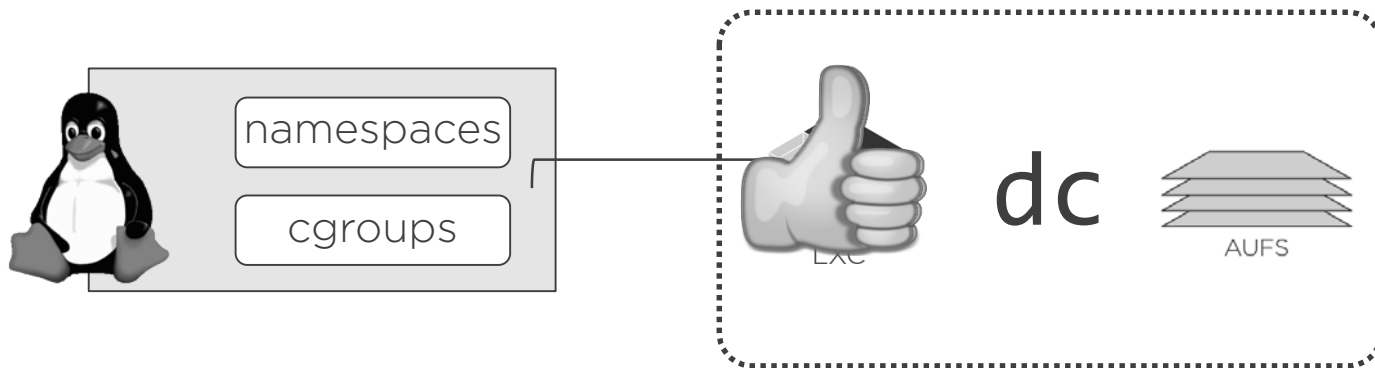


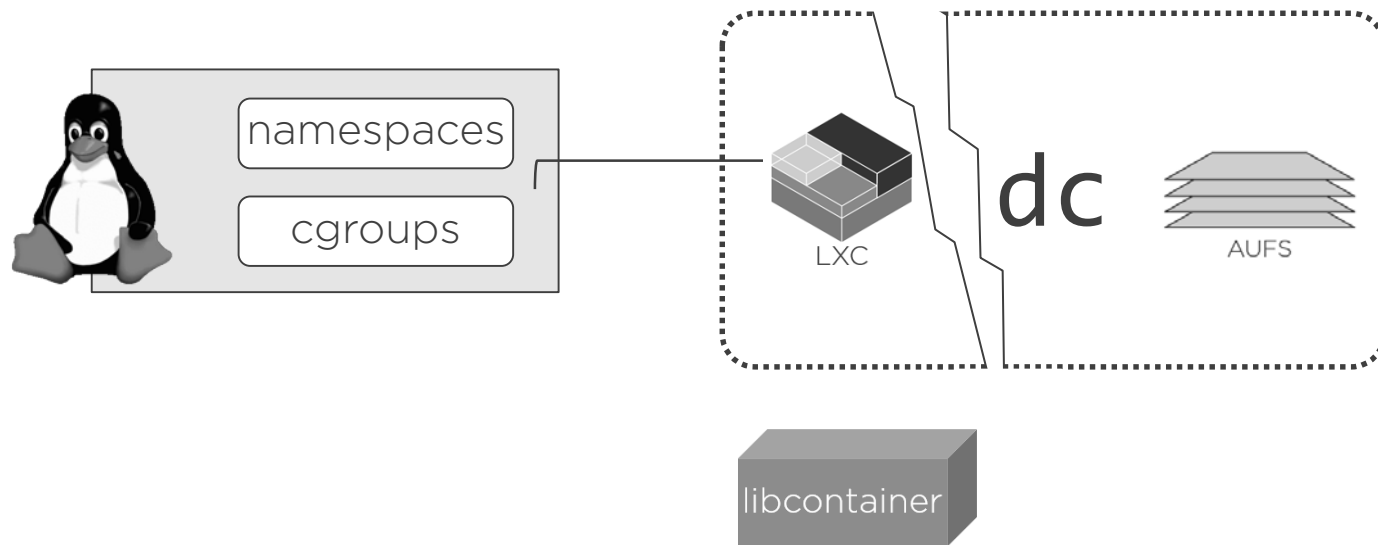
dotCloud



dc

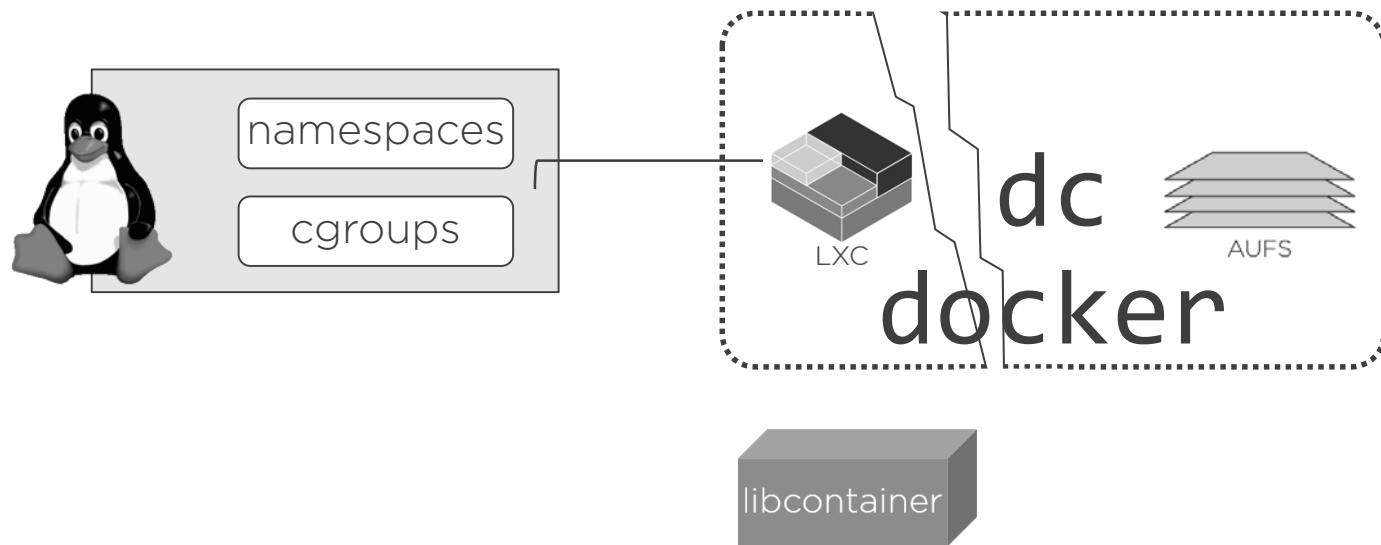






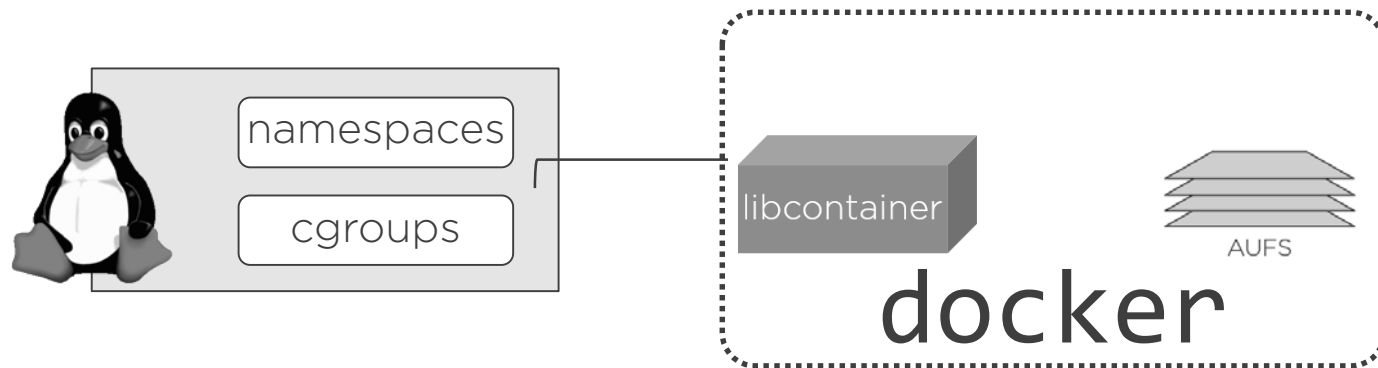
“libcontainer” (in case the diagram is too small)





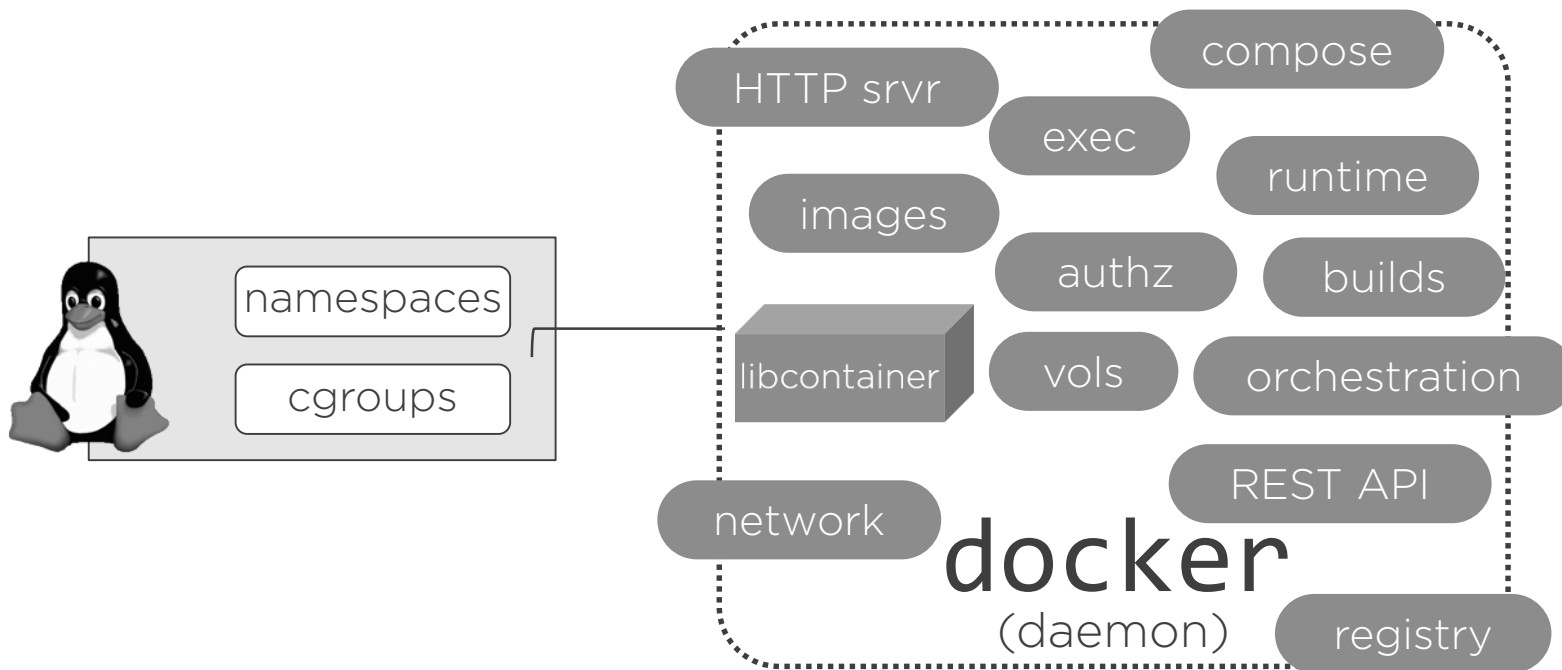
“libcontainer” (in case the diagram is too small)

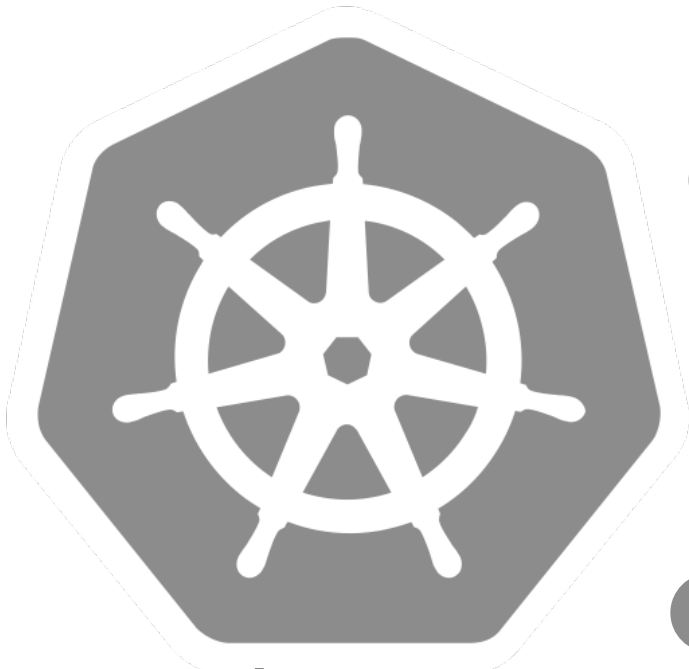




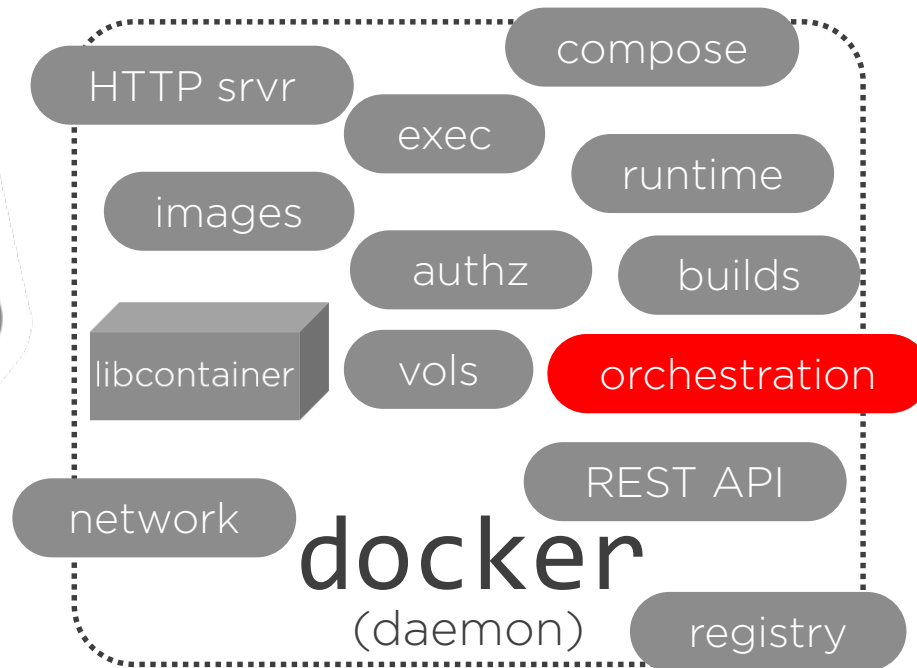
“libcontainer” (in case the diagram is too small)

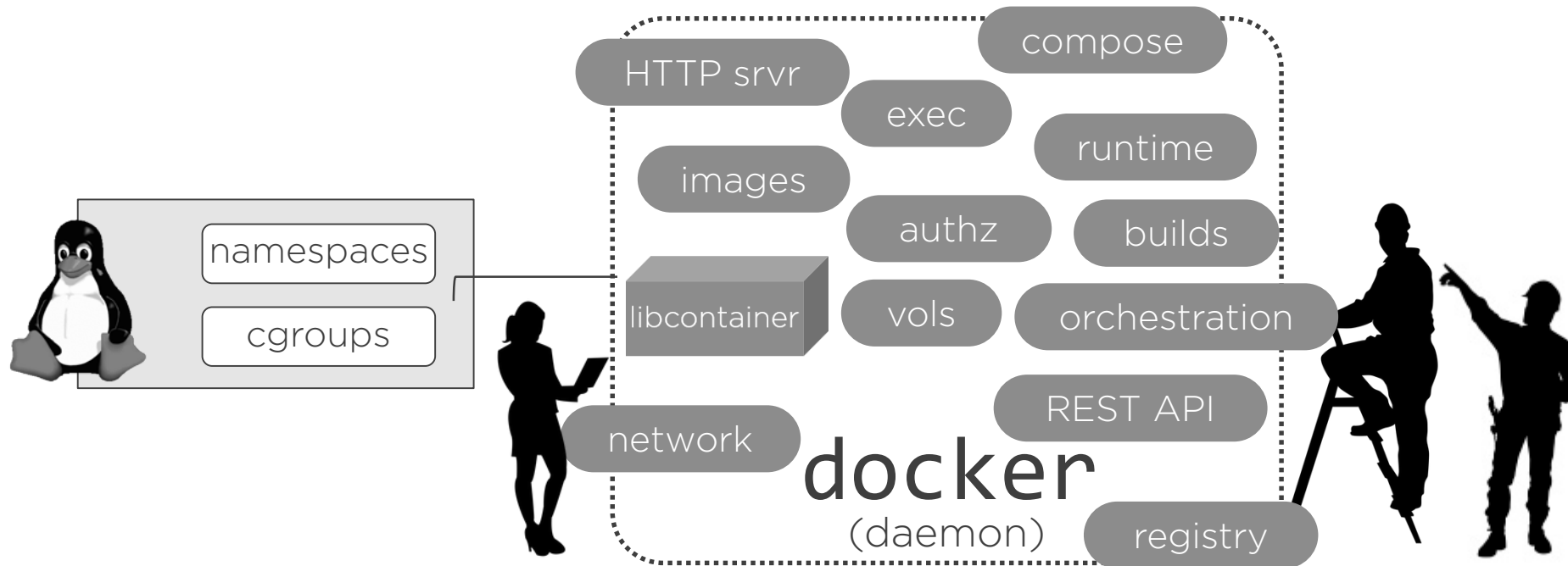


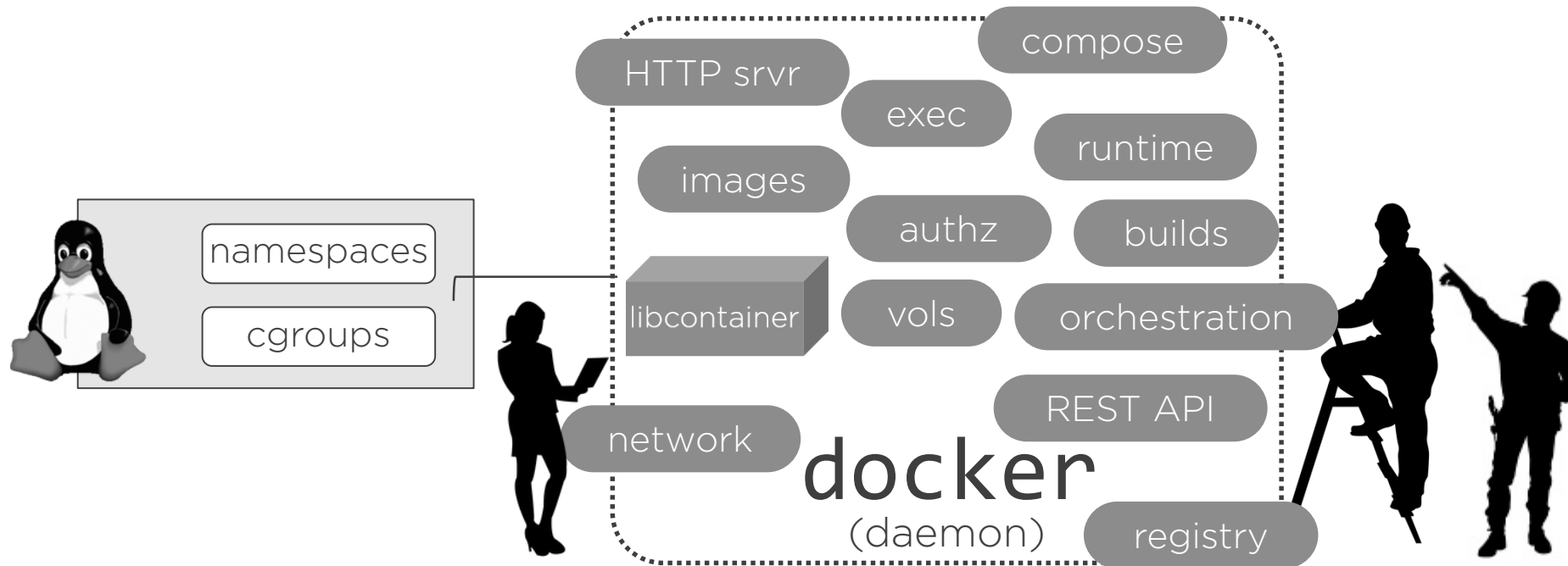


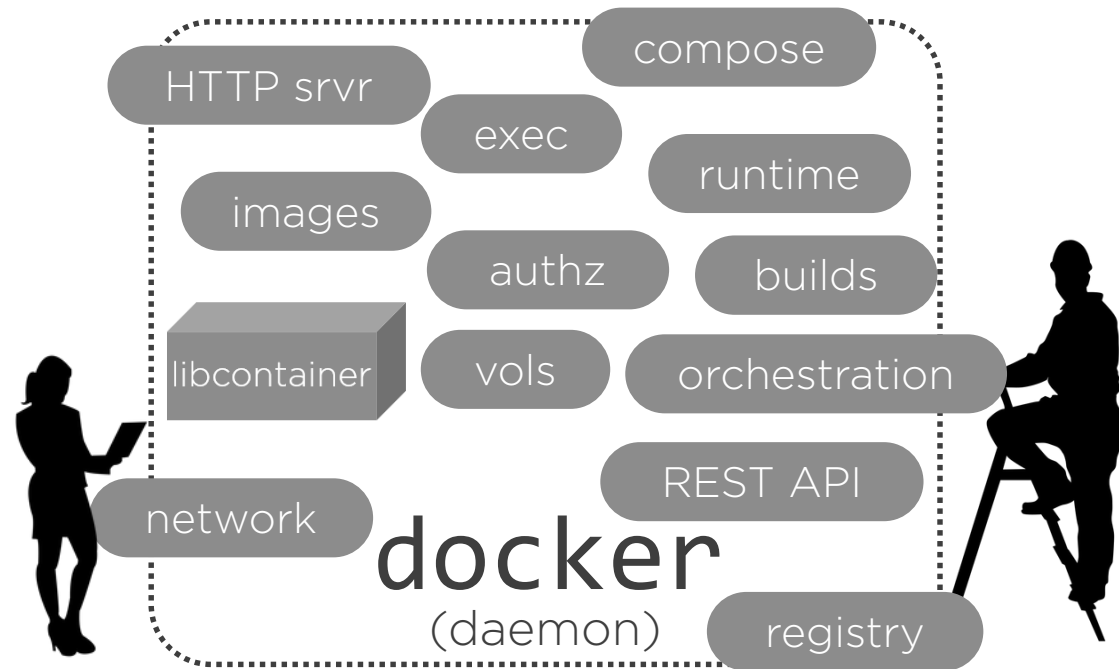


Kubernetes
(container **orchestrator**)





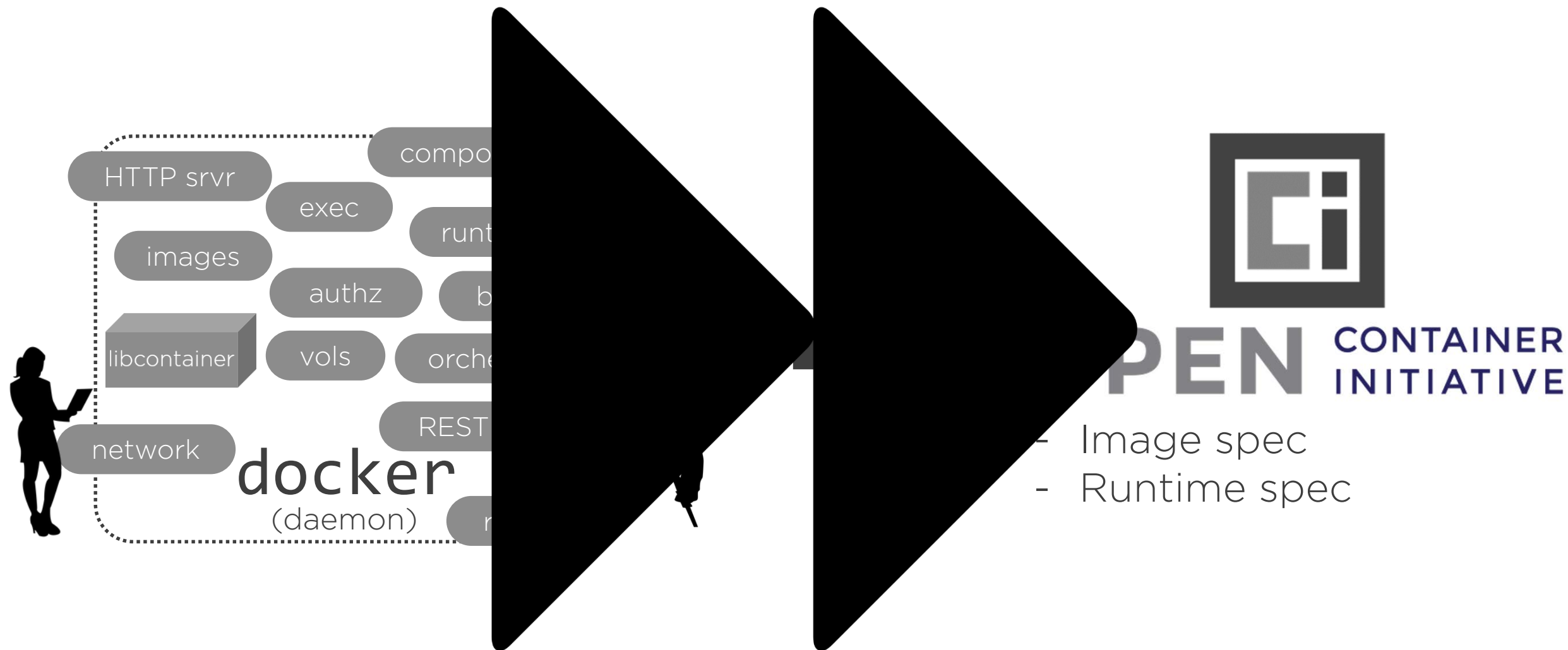


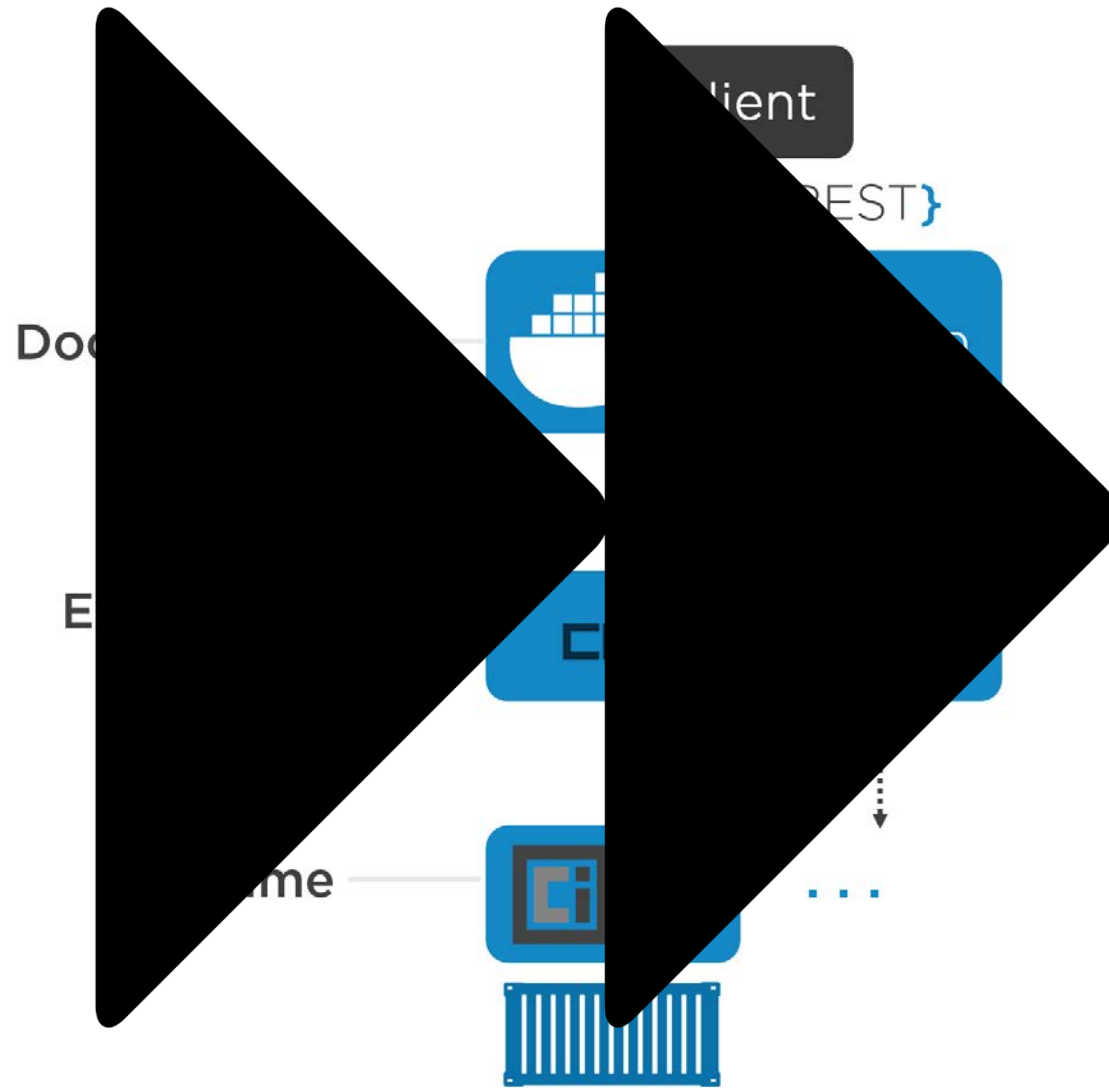


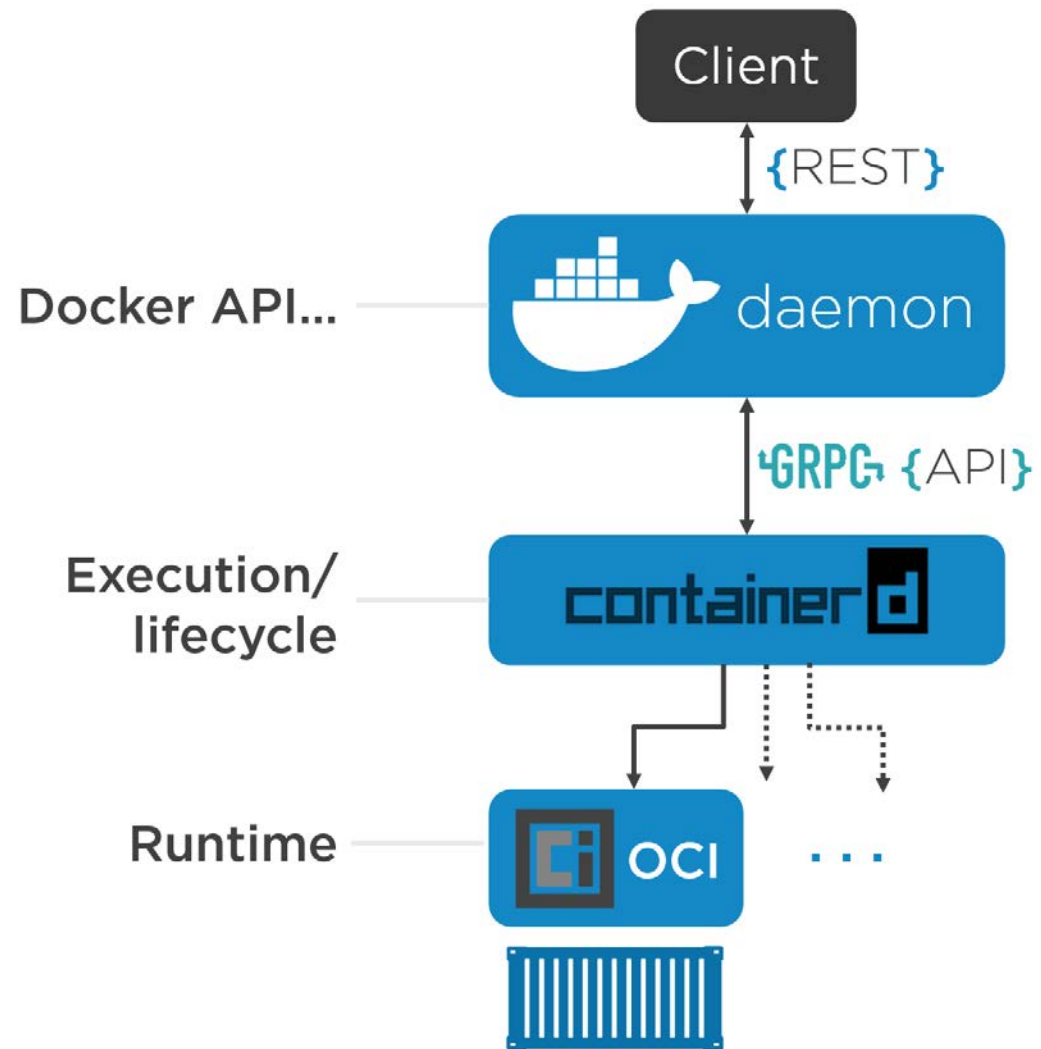
OPEN CONTAINER
INITIATIVE

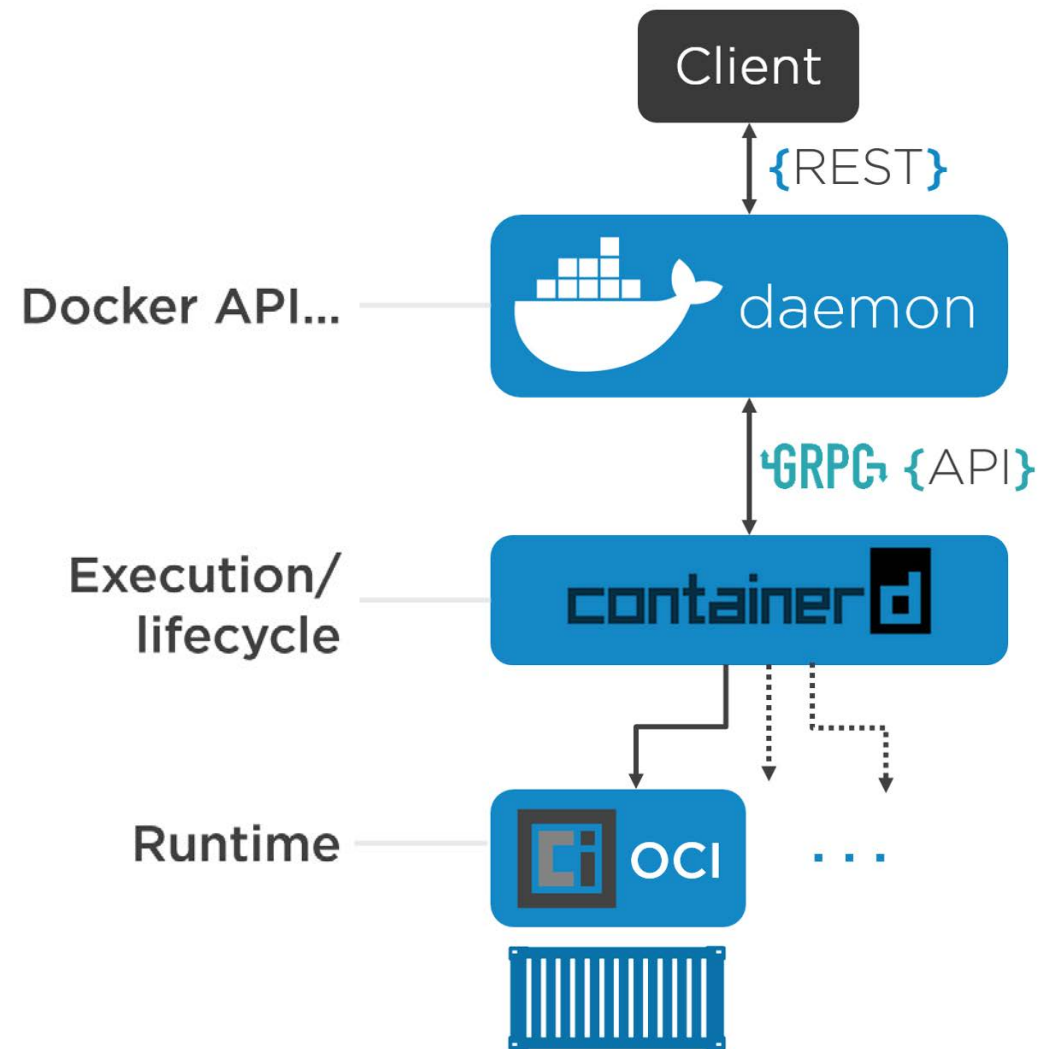
- Image spec
- Runtime spec

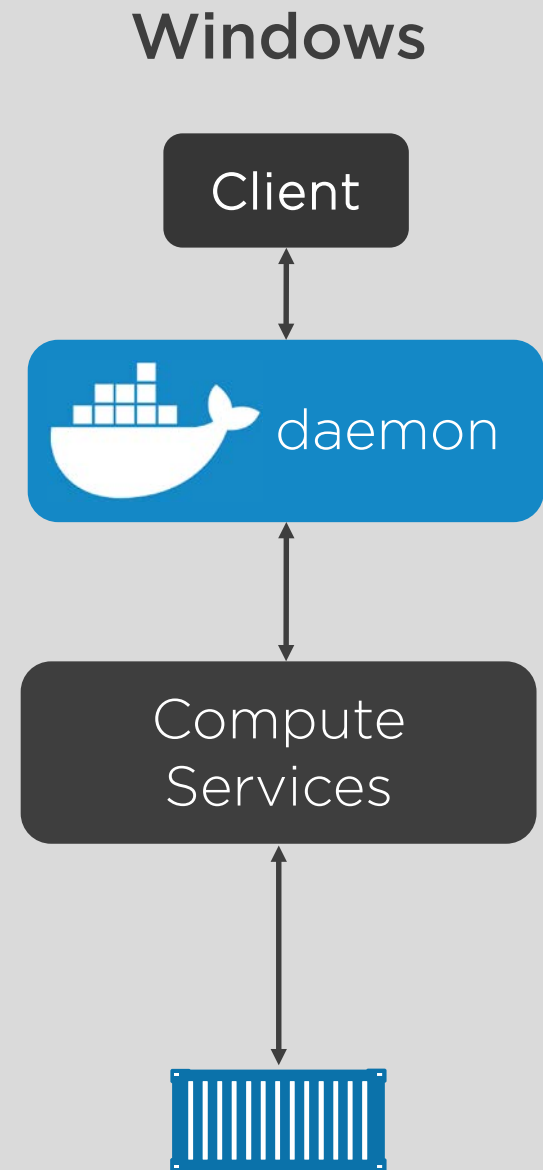
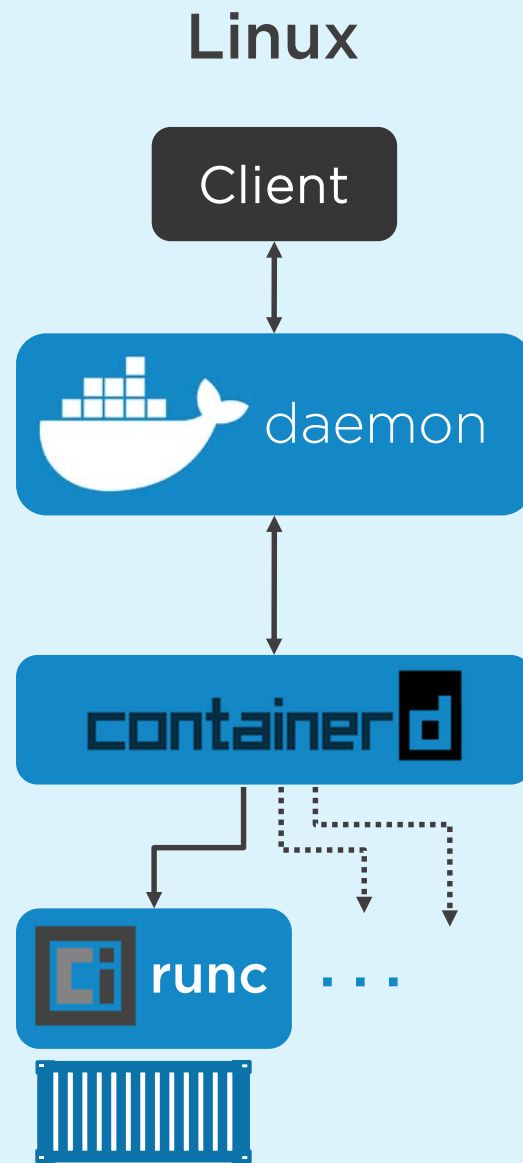
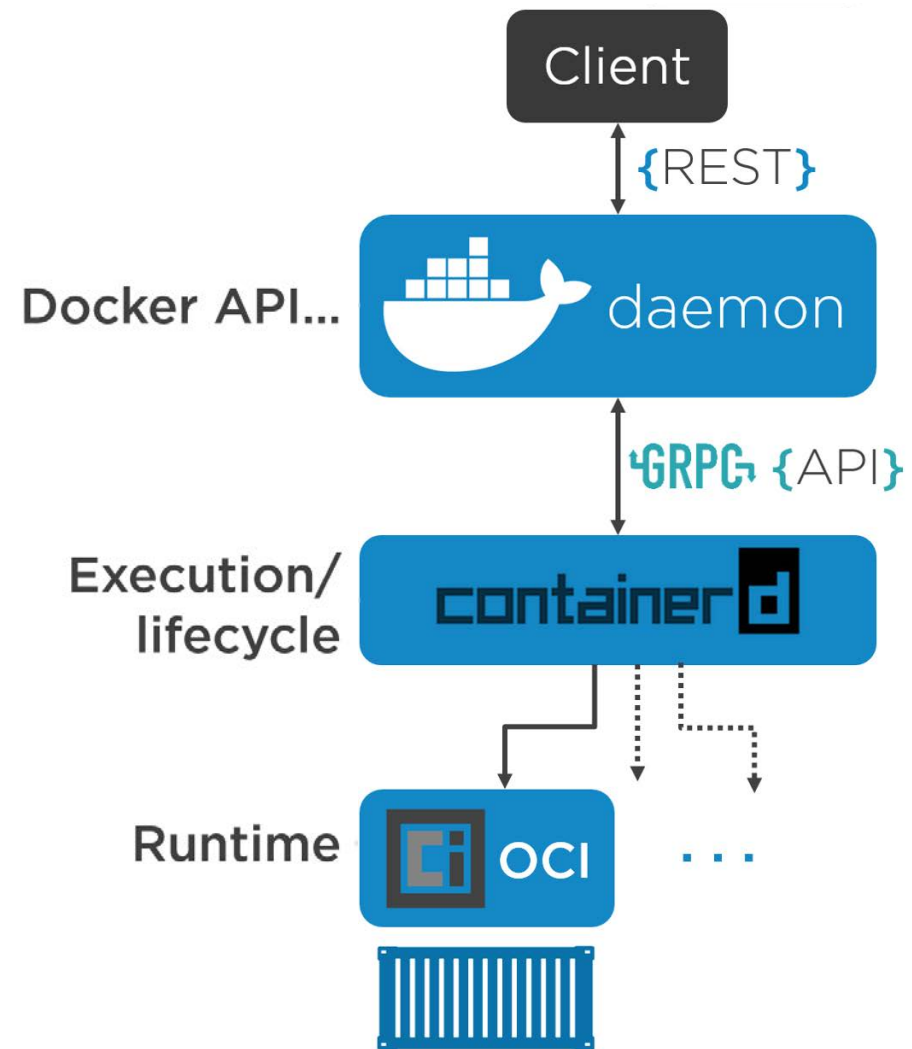




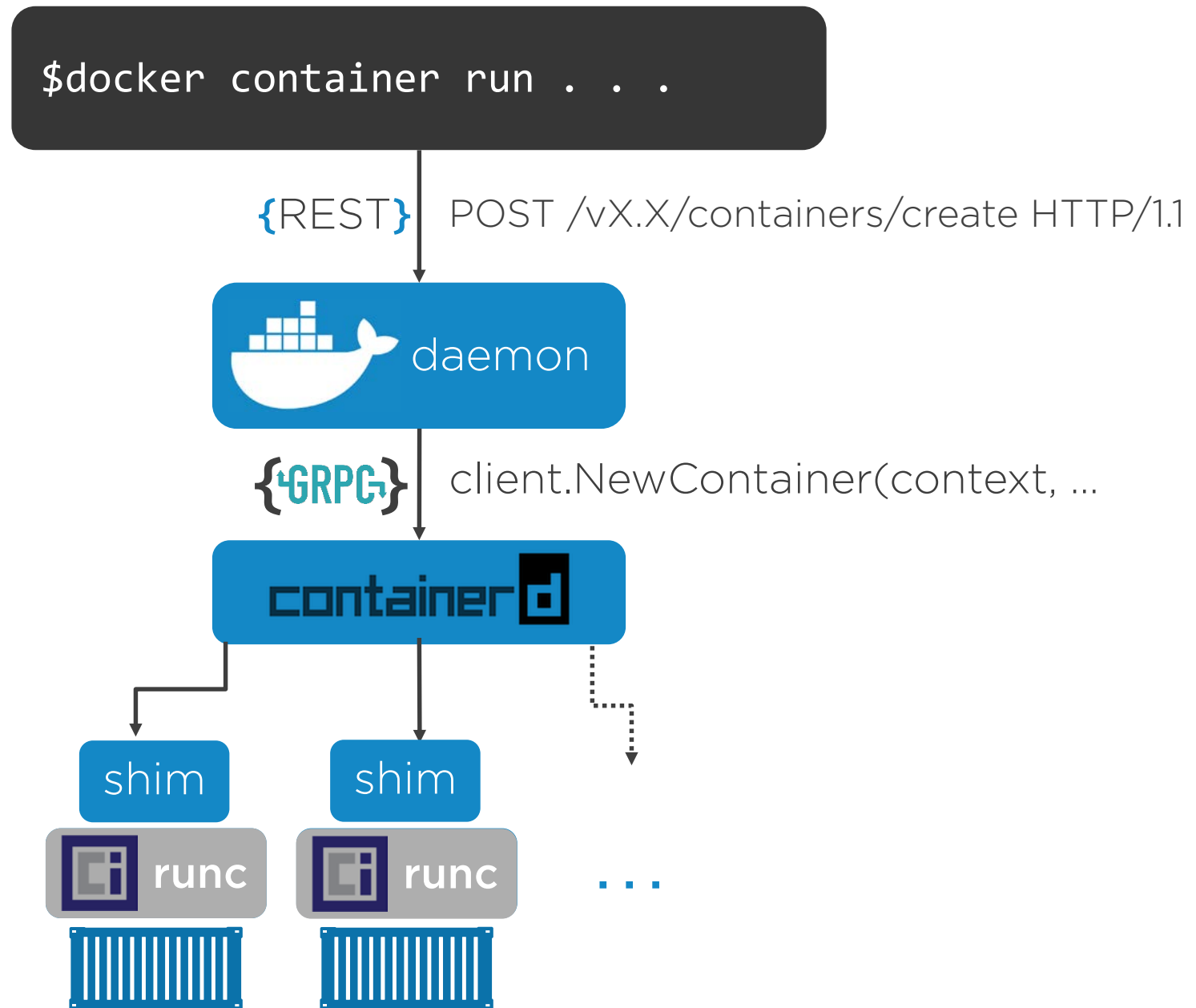






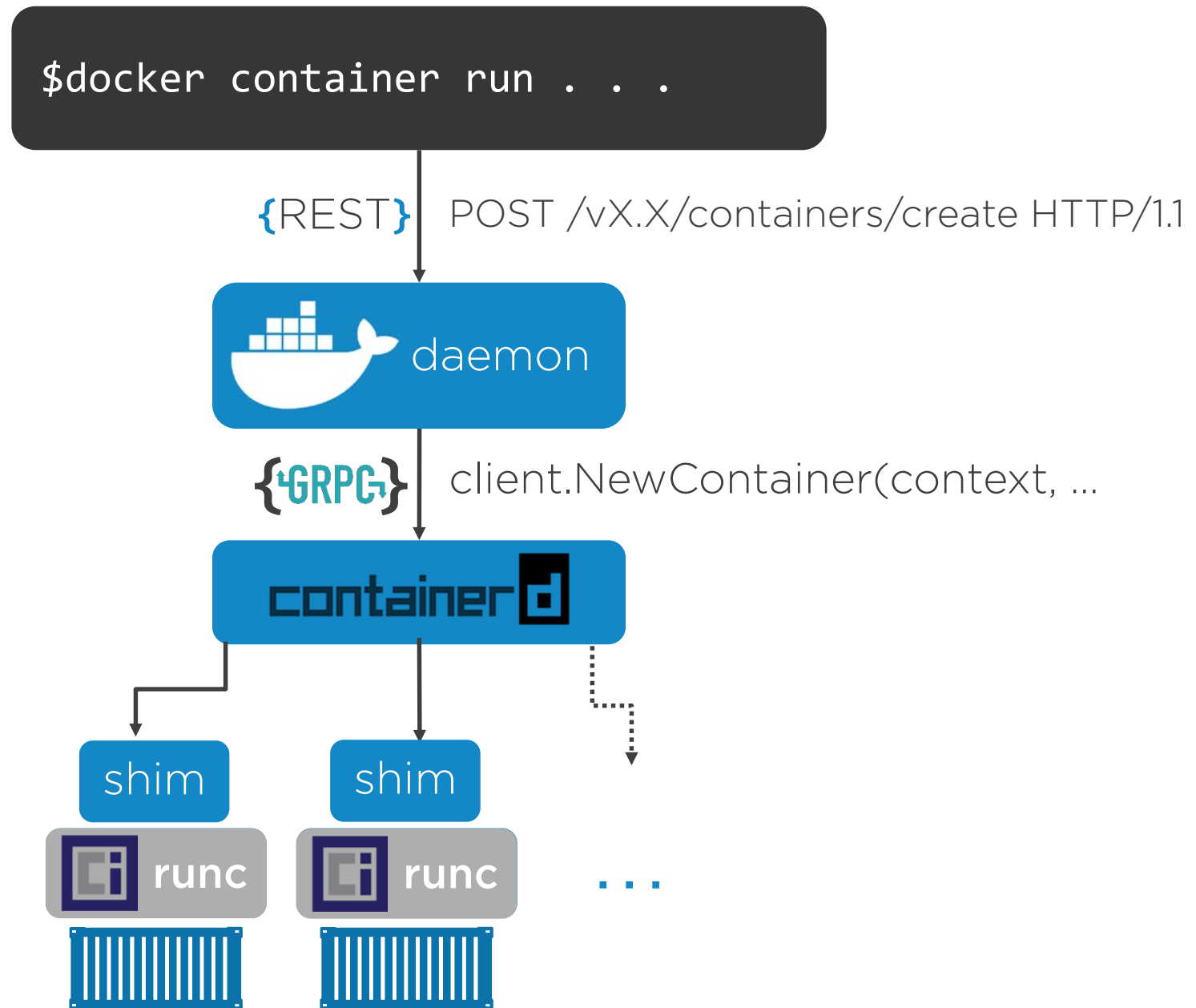


Example:
Creating a new
container on
Linux

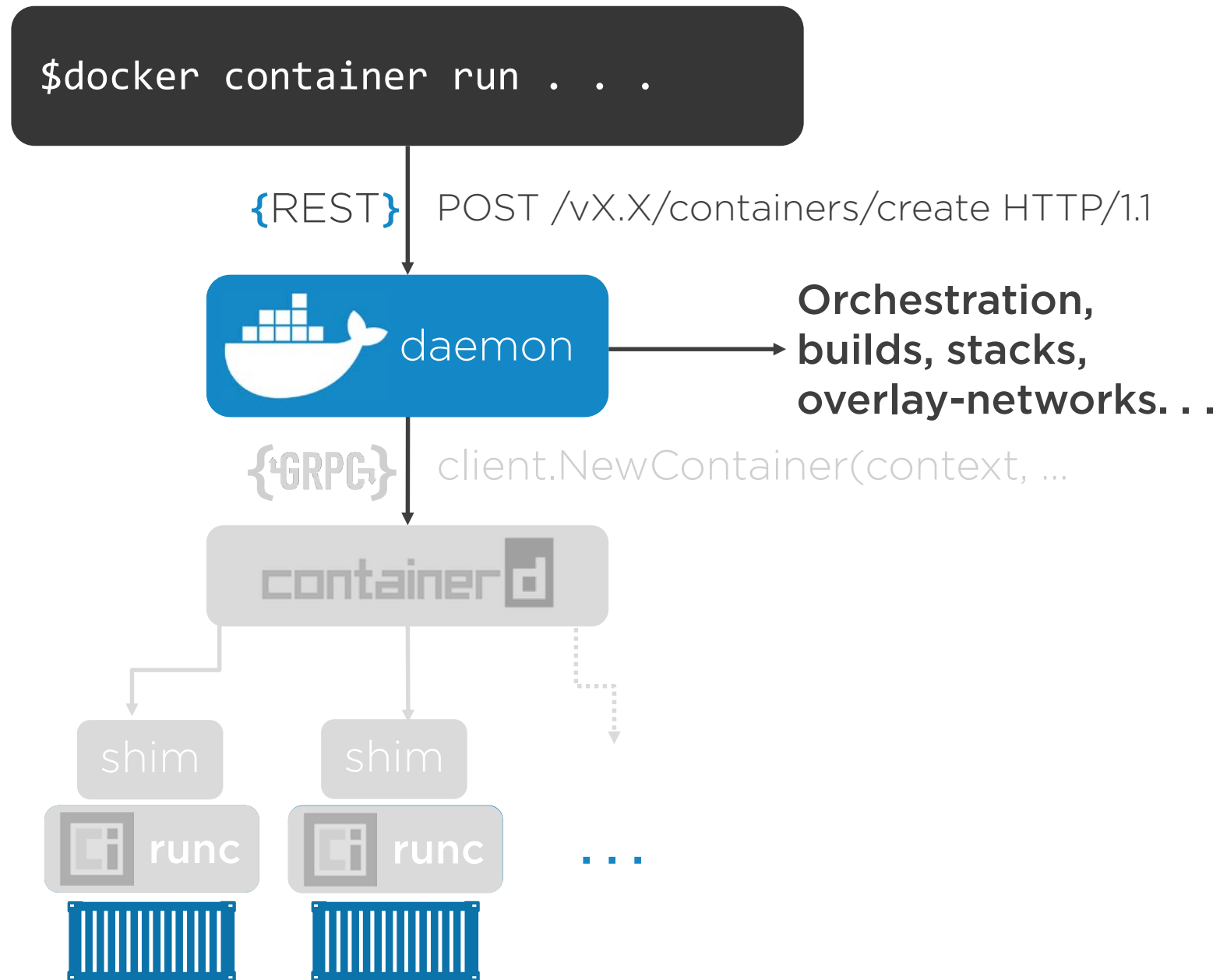


Example:

Creating a new container on Linux

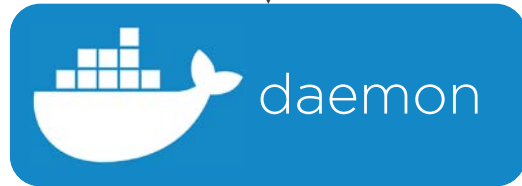


Example:
Creating a new
container on
Linux



```
$docker container run . . .
```

{REST}



Orchestration,
builds, stacks,
overlay-networks. . .

{GRPC}



shim

shim



. . .

There's a branch of **containerd** that's implementing things like:

- Basic networking
- Basic storage
- Basic image distribution
- ...

Each one is well-defined and not tightly coupled.



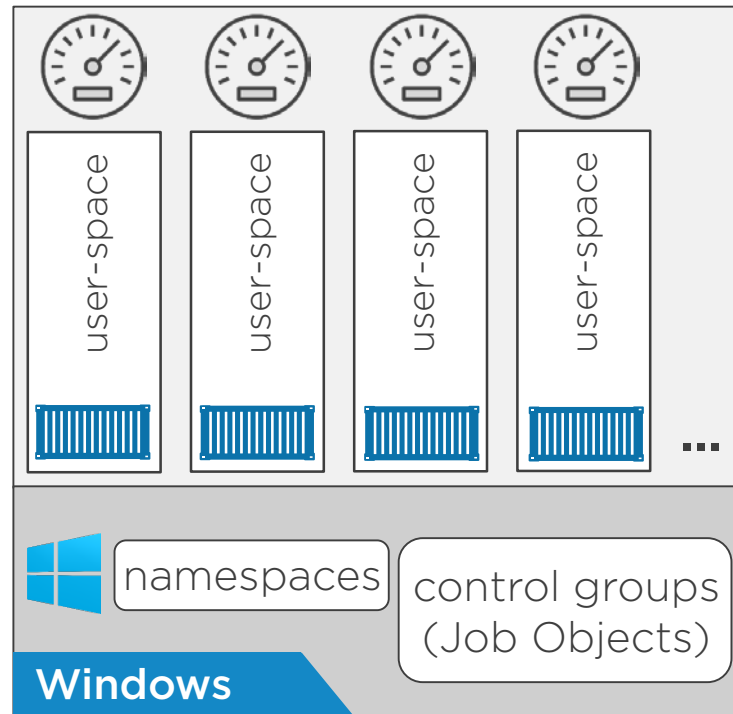
Coming up
Docker on Windows

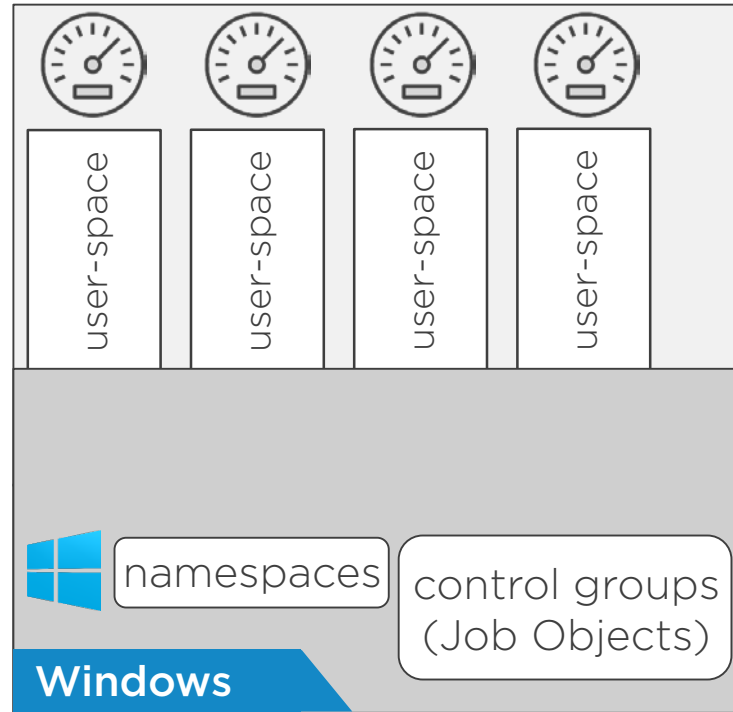


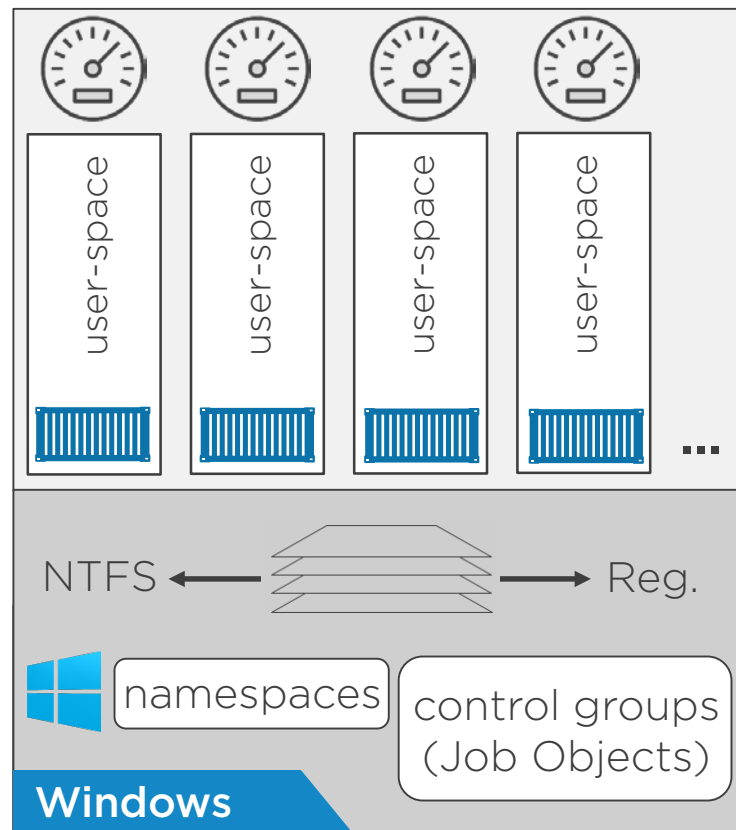
Docker on Windows

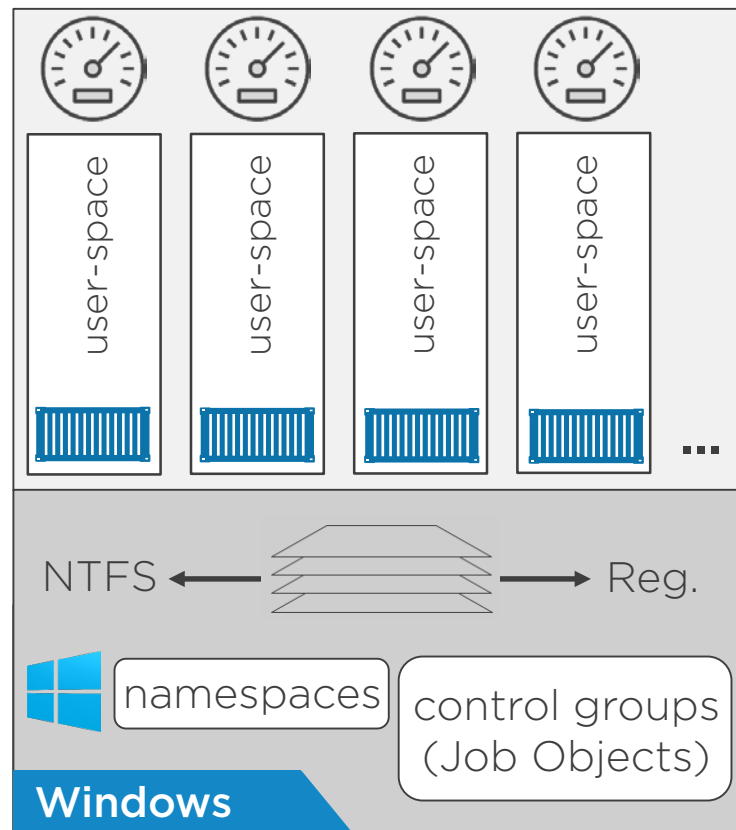
Catching up fast

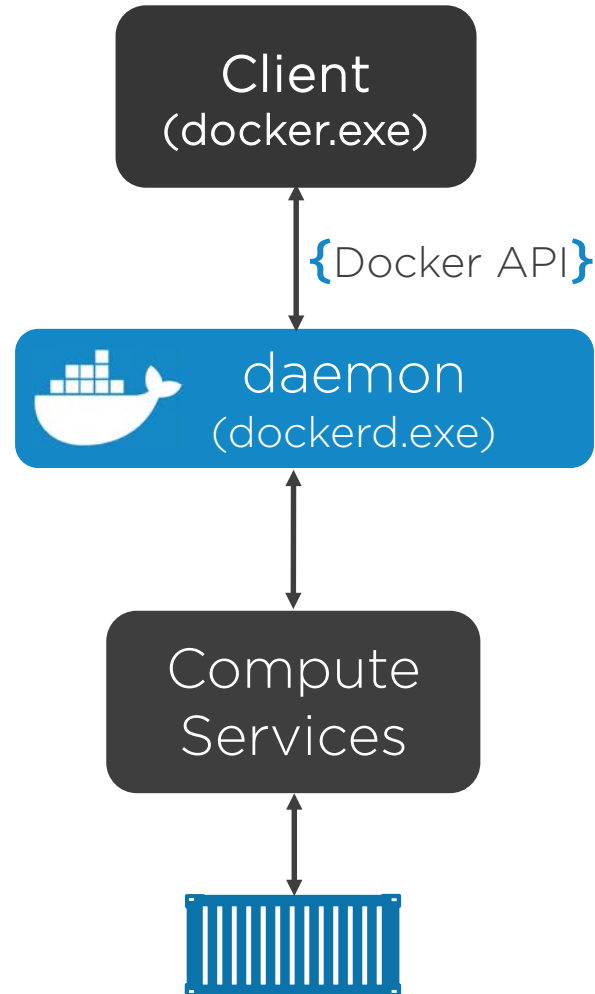
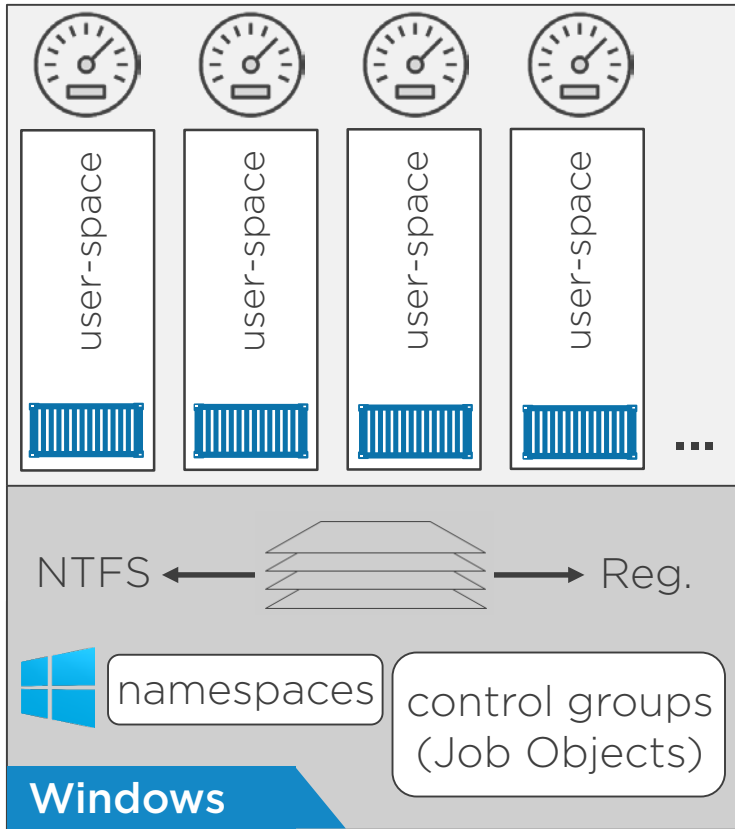


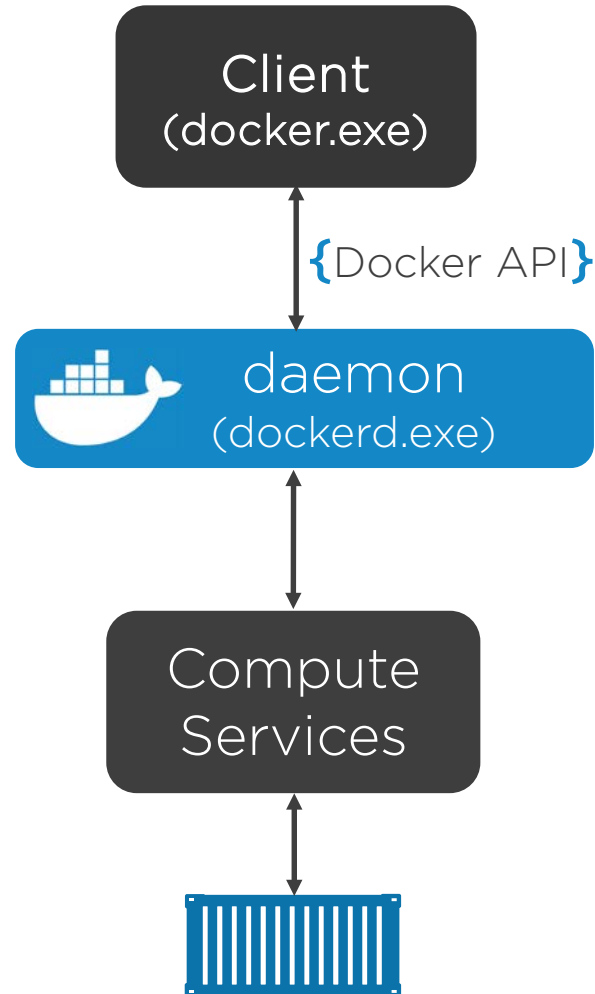
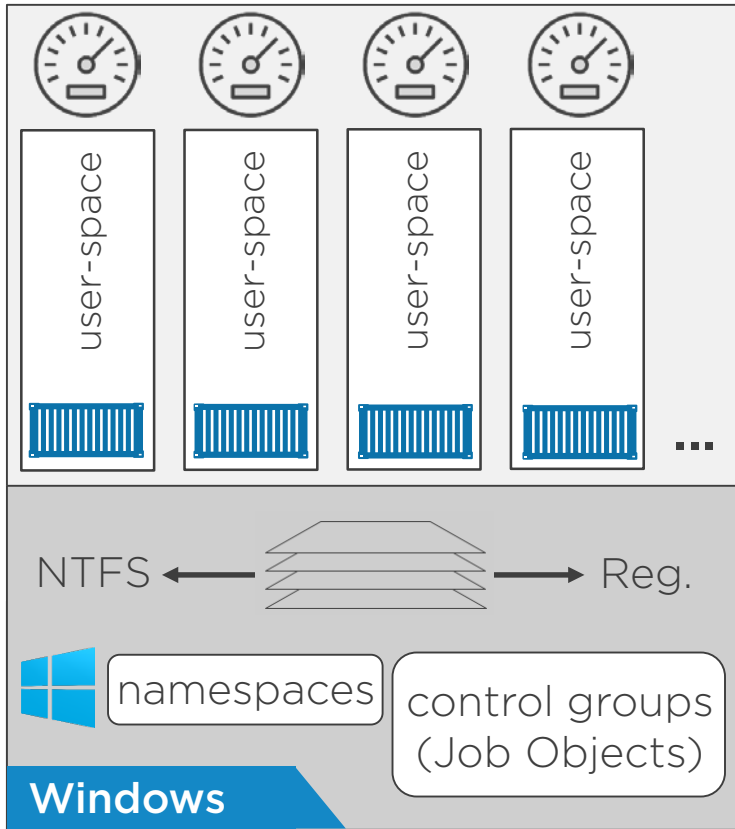


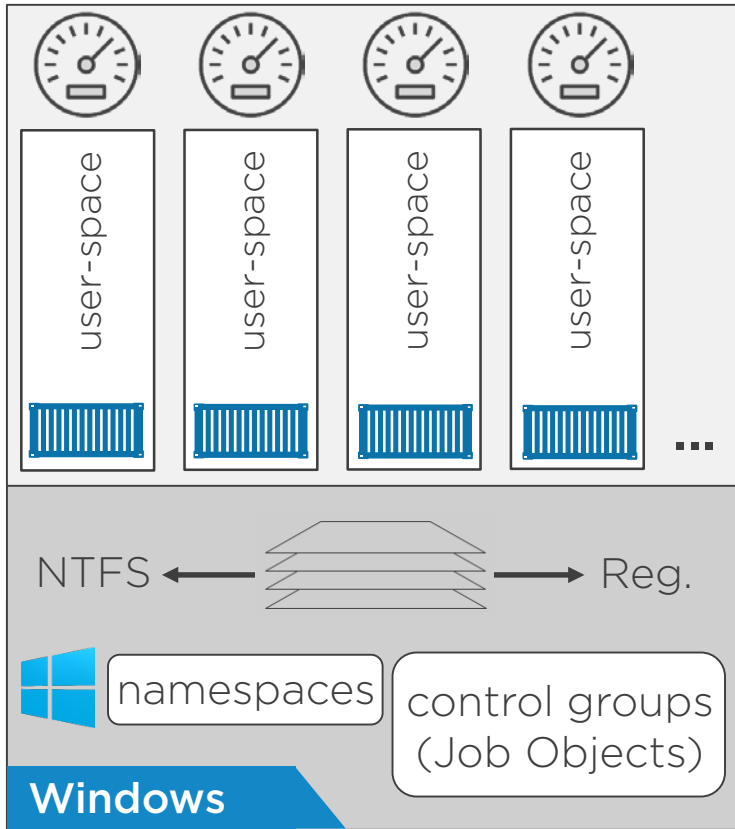






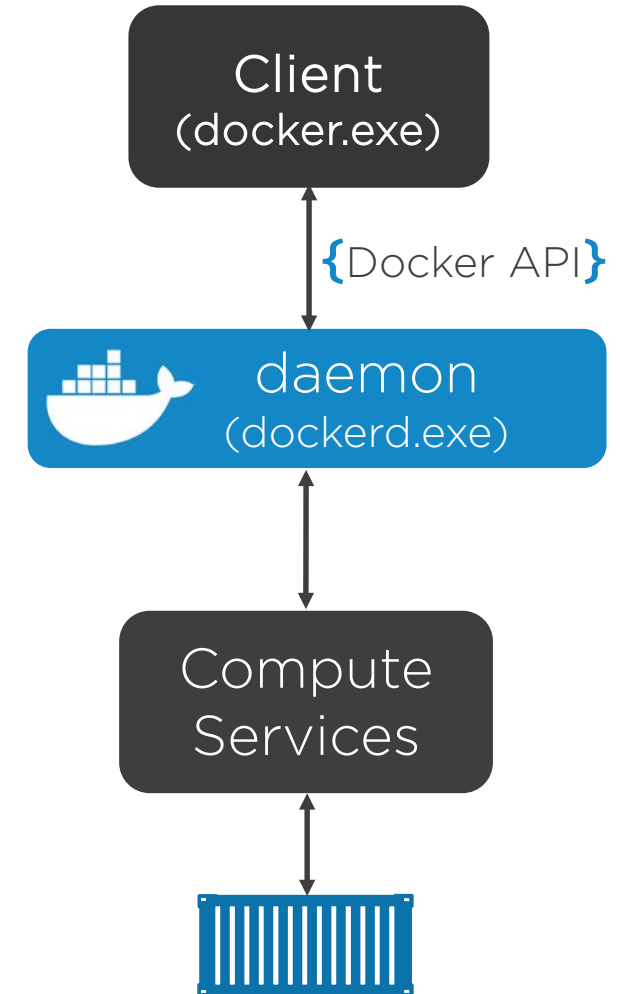


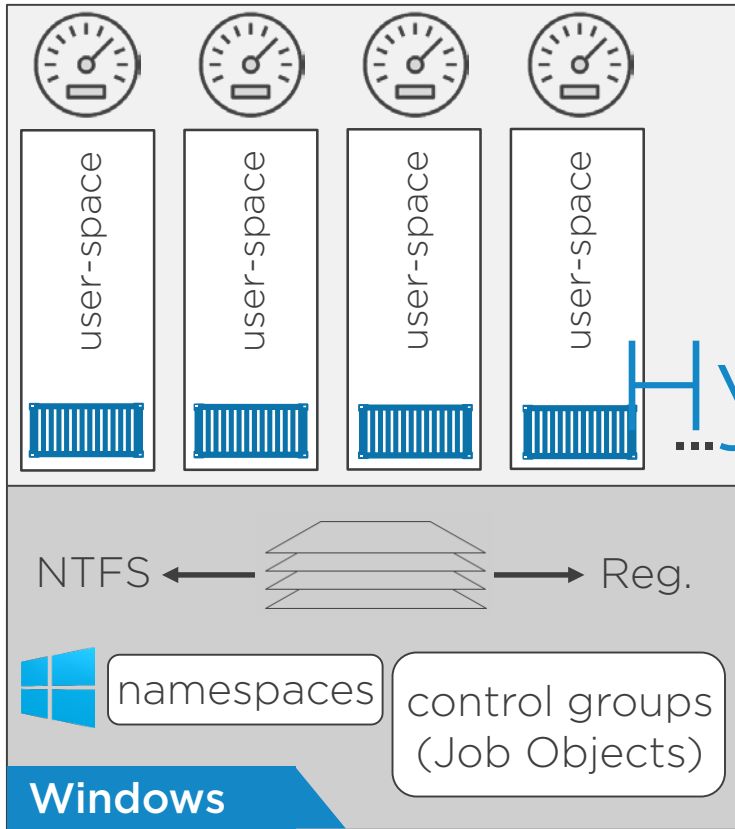




```

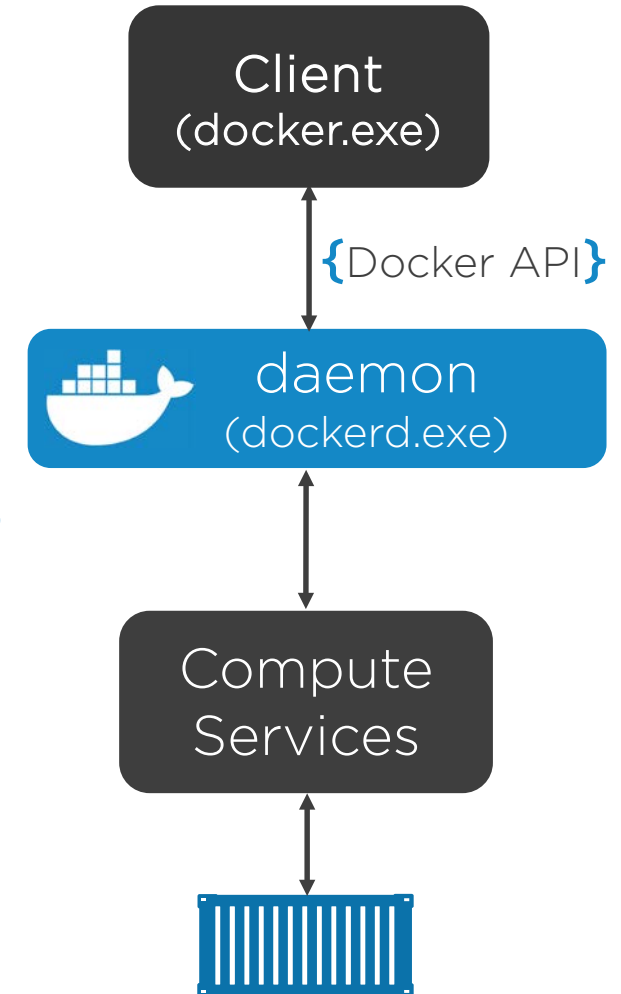
PS C:\> ps
Handles  NPM(K)  PM(K)  WS(K) CPU(s)  Id  SI ProcessName
-----
0         5    964   1292   0.00  4716  4 CExecSvc
0         5    592    956   0.00  4524  4 csrss
0         0      0      4      0.00    0  0 Idle
0        18   3984   8624   0.13   700  4 lsass
0       52  26624  19400   1.64  2100  4 powershell
0       38  28324  49616   1.69  4464  4 powershell
0         8   1488   3032   0.06  2488  4 services
0         2    288    504   0.00  4508  0 smss
0         8   1600   3004   0.03   908  4 svchost
0        12   1492   3504   0.06  4572  4 svchost
0        15  20284  23428   5.64  4628  4 svchost
0        15   3704   7536   0.09  4688  4 svchost
0        28   5708   6588   0.45  4712  4 svchost
0        10   2028   4736   0.03  4840  4 svchost
0        11   5364   4824   0.08  4928  4 svchost
0         0    128    136  37.02    4  0 System
0         7    920   1832   0.02  3752  4 wininit
0         8   5472  11124   0.77  5568  4 WmiPrvSE
  
```

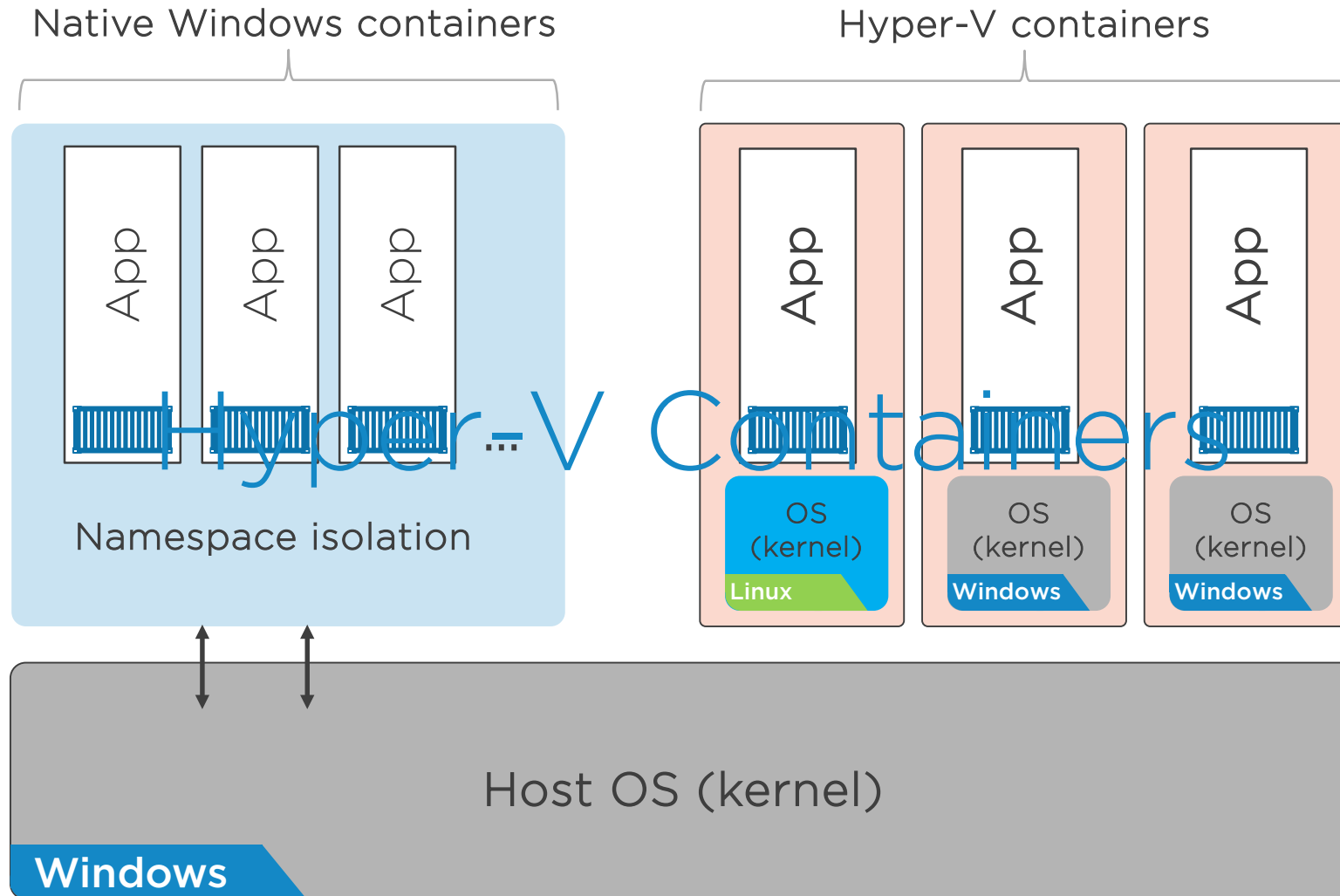




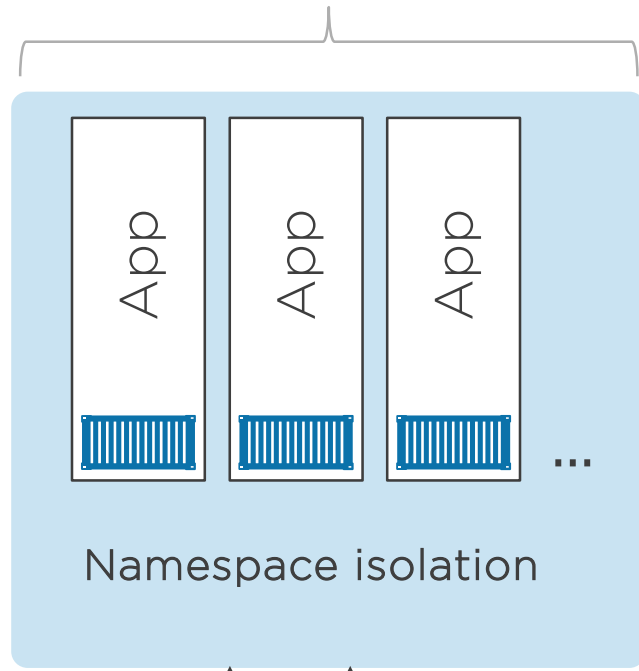
```

PS C:\> ps
Handles NPM(K) PM(K) WS(K) CPU(s) Id SI ProcessName
-----
0 5 964 1292 0.00 4716 4 CExecSvc
0 5 592 956 0.00 4524 4 csrss
0 0 0 4 0 0 Idle
0 18 3984 8624 0.13 700 4 lsass
0 52 26624 19400 1.64 2100 4 powershell
0 33 28324 49616 1.69 4464 4 powershell
0 8 1488 3032 0.06 2488 4 services
0 2 288 504 0.00 4508 0 smss
0 8 1600 3004 0.03 908 4 svchost
0 12 1492 3504 0.06 4572 4 svchost
0 15 20284 23428 5.64 4628 4 svchost
0 15 3704 7536 0.09 4688 4 svchost
0 28 5708 6588 0.45 4712 4 svchost
0 10 2028 4736 0.03 4840 4 svchost
0 11 5364 4824 0.08 4928 4 svchost
0 0 128 136 37.02 4 0 System
0 7 920 1832 0.02 3752 4 wininit
0 8 5472 11124 0.77 5568 4 WmiPrvSE
  
```

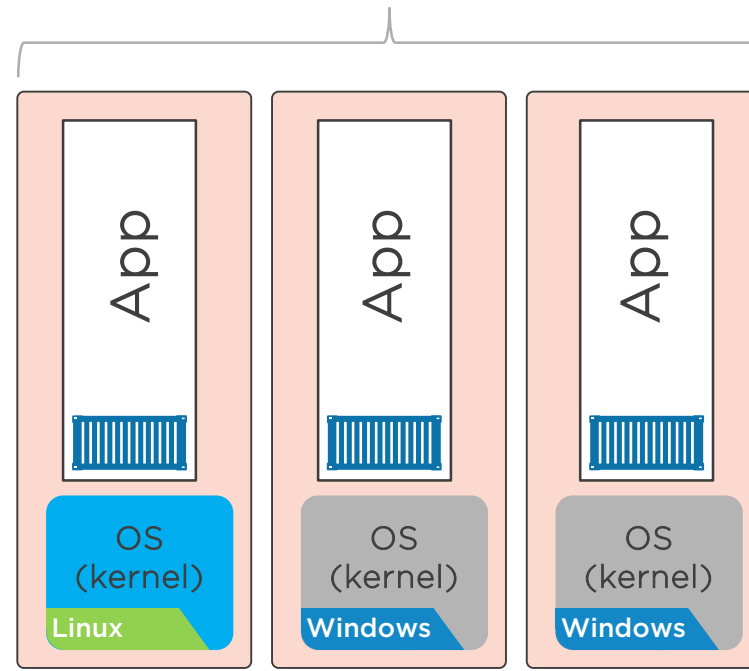




Native Windows containers



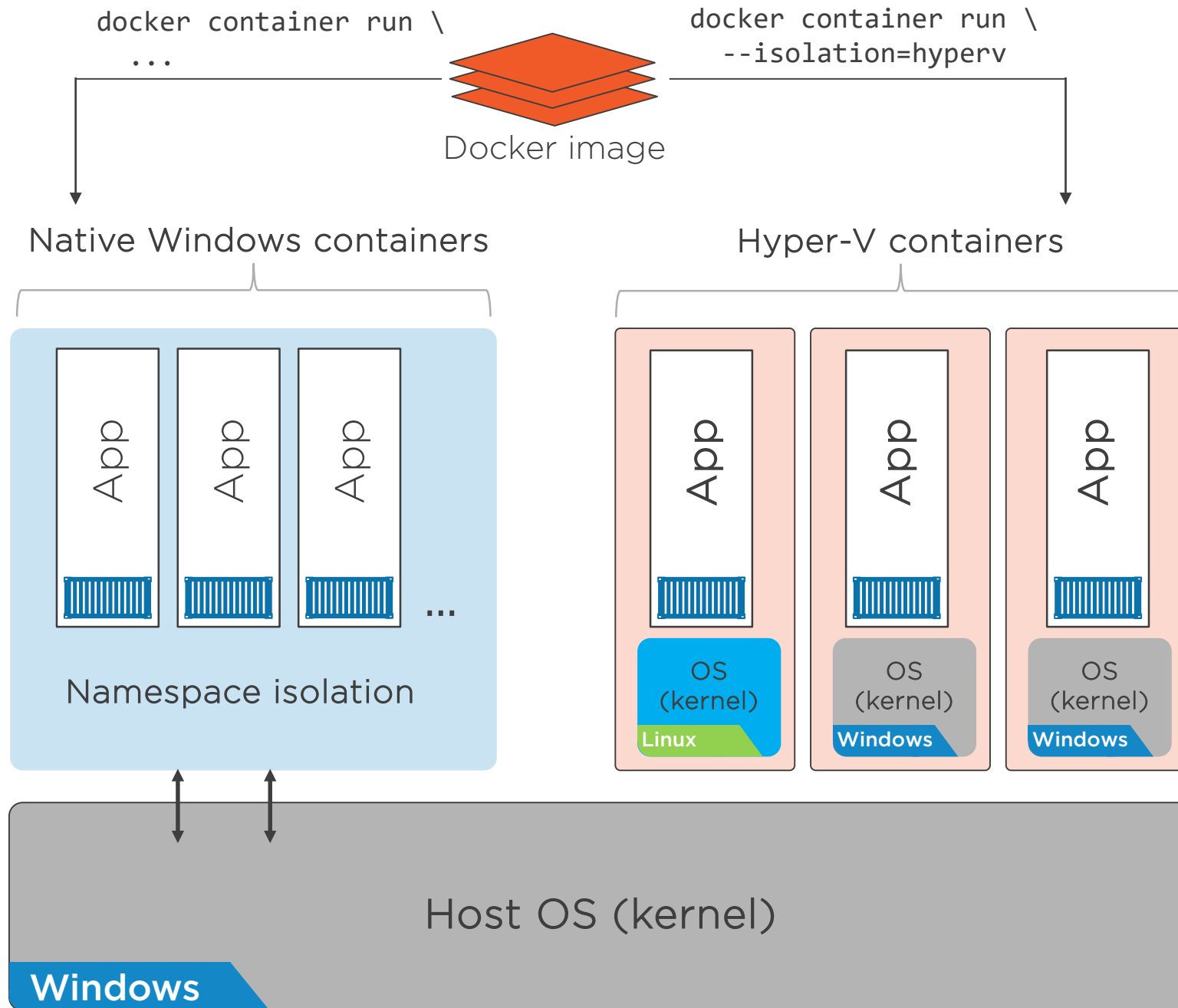
Hyper-V containers



Host OS (kernel)

Windows





Coming up
Uber recap!



Recap

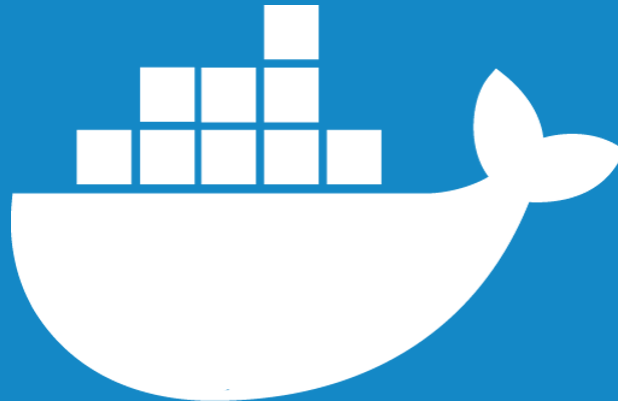
Reminding you of everything you've already forgotten

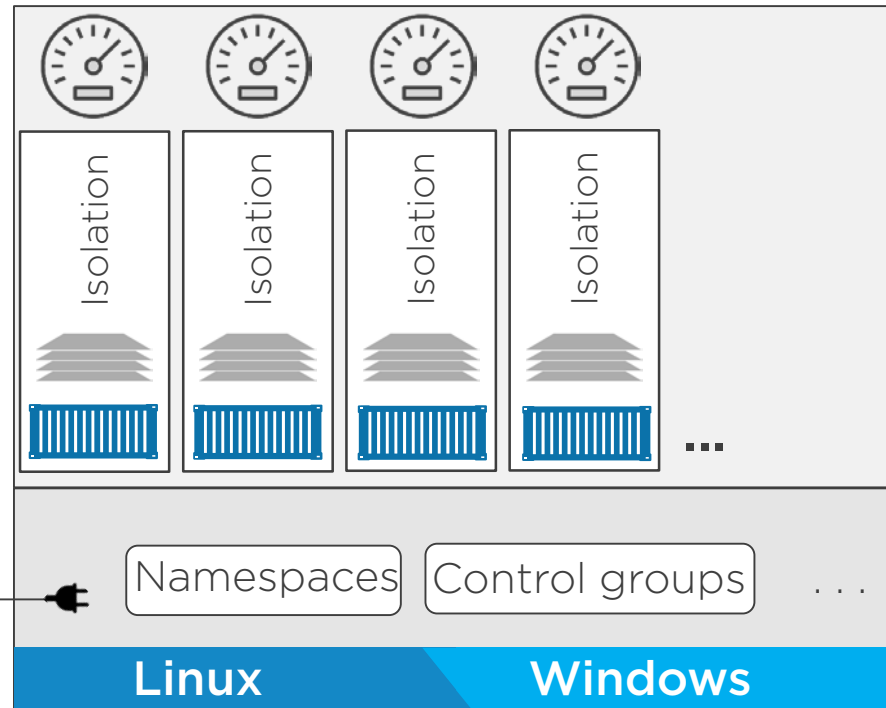
Docker on Linux

Where it all started

Docker on Windows

Playing catch-up



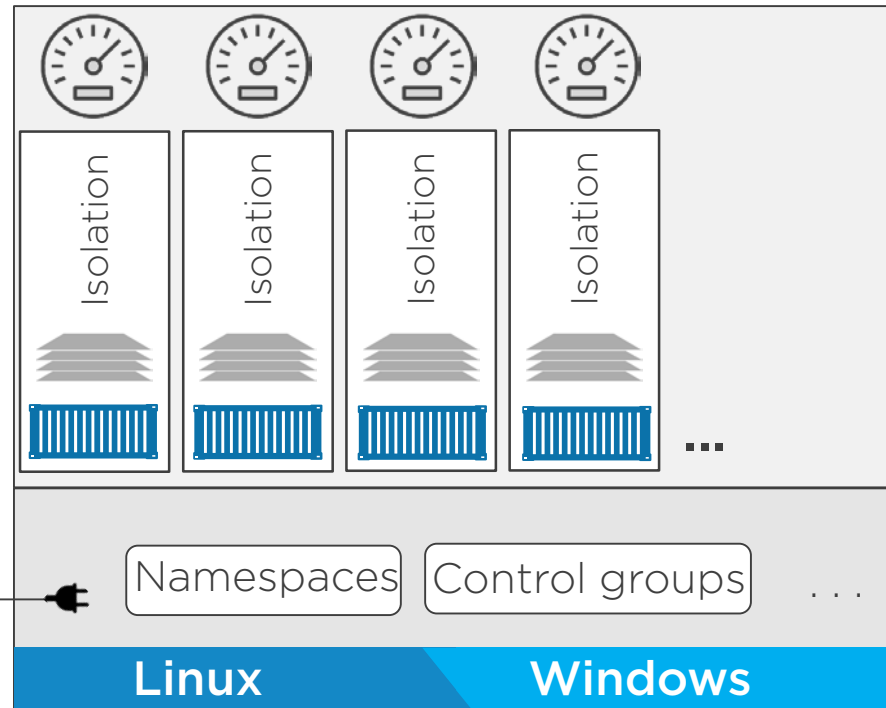


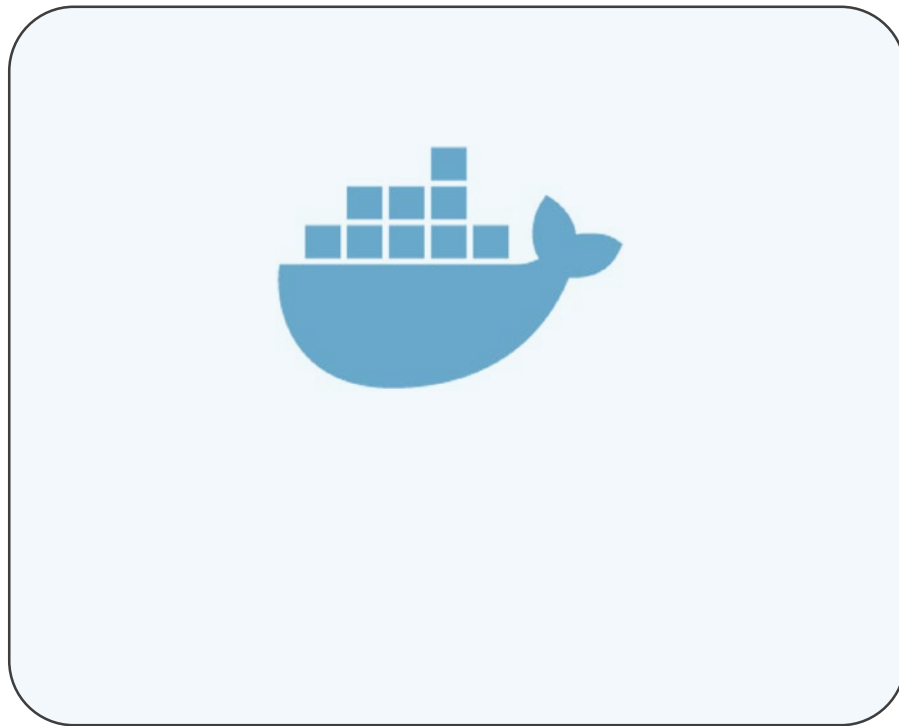
Layered filesystem/
snapshot & CoW

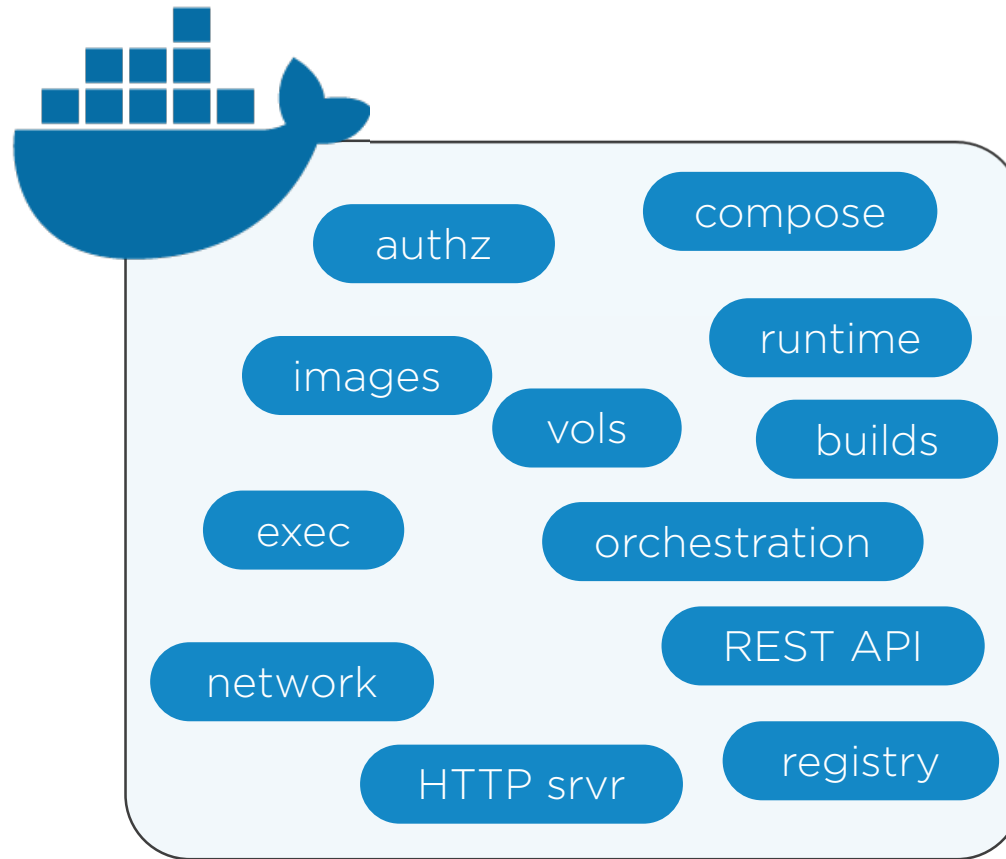




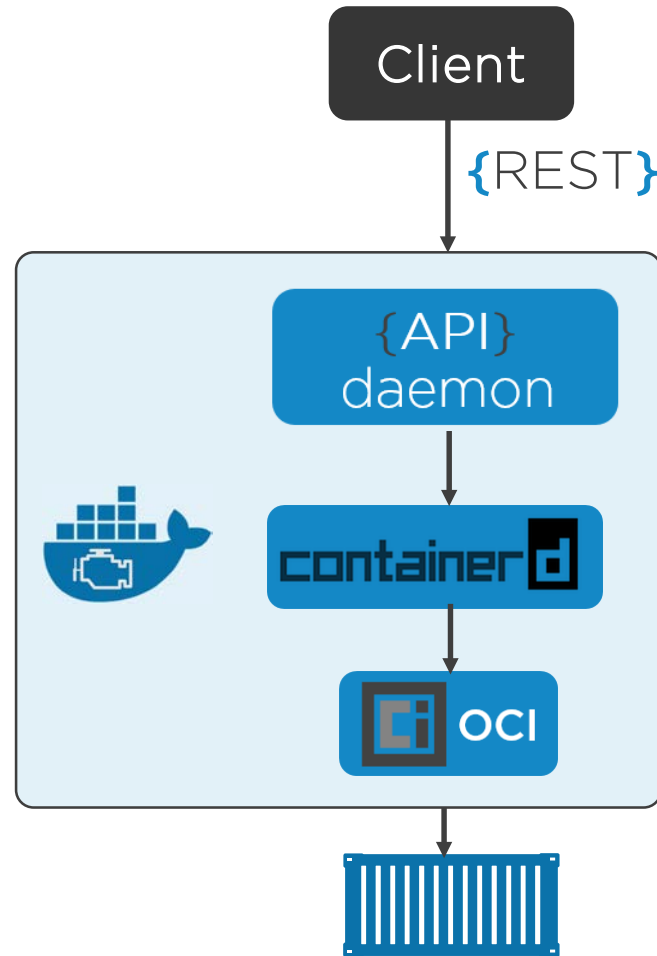
Layered filesystem/
snapshot & CoW



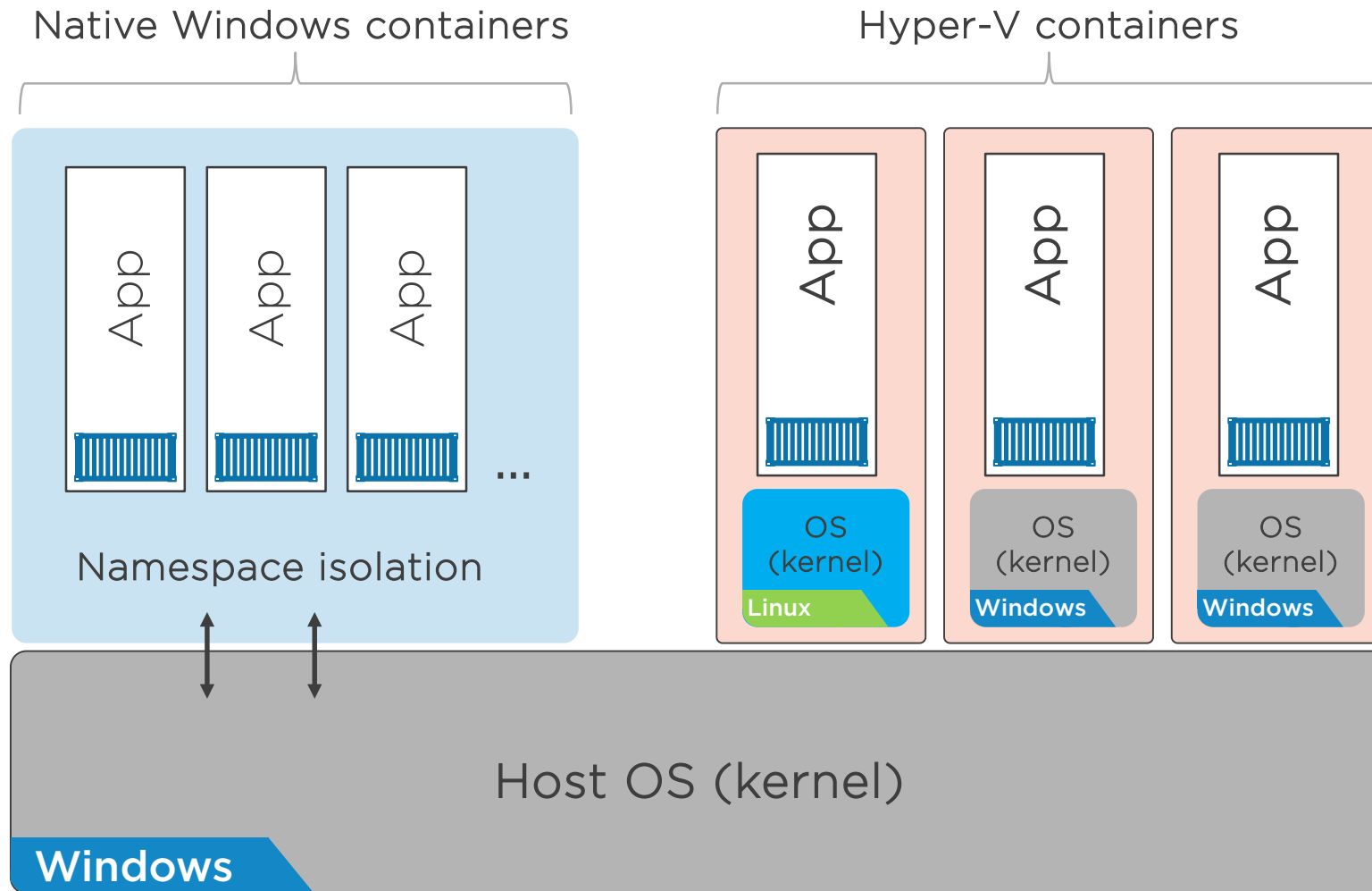




Some diagrams show runc *inside* of containerd. This is because containerd manages runc and both are loosely *runtimes*







Coming up

NEXT MODULE

Working with Images

