



# DEMYSTIFYING DOCKER



PRASHANTH

CLOUD PLATFORM, SAP

# AGENDA

- Linux Internals
  - Processes
  - Memory & Storage
- How does OS help in achieving
  - Resource control
  - Isolation
  - Security
- Docker
  - Overview
  - Usage

```

ile.c
libtool: compile: x86_64-pc-linux-gnu-gcc -DHAVE_CONFIG_H -I. -I../../include -I../../../include -I./ref -I/usr/include/et -pipe -O2 -march=native -D_LARGE_FILES= -Wall -Wmissing-prototypes -Wpointer-arith -Wbad-function-cast -Wmissing-declarations -Wnested-externs -pipe -O2 -march=native -c file.c -fPIC -DPIC -o .libs/libhx509_la-file.o
/bin/sh ../../libtool --tag=CC --mode=compile x86_64-pc-linux-gnu-gcc -DHAVE_CONFIG_H -I. -I../../include -I../../../include -I./ref -I/usr/include/et -pipe -O2 -march=native -D_LARGE_FILES= -Wall -Wmissing-prototypes -Wpointer-arith -Wbad-function-cast -Wmissing-declarations -Wnested-externs -pipe -O2 -march=native -c -o libhx509_la-sel.lo `test -f 'sel.c' || echo './sel.c'
libtool: compile: x86_64-pc-linux-gnu-gcc -DHAVE_CONFIG_H -I. -I../../include -I../../../include -I./ref -I/usr/include/et -pipe -O2 -march=native -D_LARGE_FILES= -Wall -Wmissing-prototypes -Wpointer-arith -Wbad-function-cast -Wmissing-declarations -Wnested-externs -pipe -O2 -march=native -c sel.c -fPIC -DPIC -o .libs/libhx509_la-sel.o
/bin/sh ../../libtool --tag=CC --mode=compile x86_64-pc-linux-gnu-gcc -DHAVE_CONFIG_H -I. -I../../include -I../../../include -I./ref -I/usr/include/et -pipe -O2 -march=native -D_LARGE_FILES= -Wall -Wmissing-prototypes -Wpointer-arith -Wbad-function-cast -Wmissing-declarations -Wnested-externs -pipe -O2 -march=native -c -o libhx509_la-sel-gram.o `test -f 'sel-gram.c' || echo './sel-gram.c'
libtool: compile: x86_64-pc-linux-gnu-gcc -DHAVE_CONFIG_H -I. -I../../include -I../../../include -I./ref -I/usr/include/et -pipe -O2 -march=native -D_LARGE_FILES= -Wall -Wmissing-prototypes -Wpointer-arith -Wbad-function-cast -Wmissing-declarations -Wnested-externs -pipe -O2 -march=native -c sel-gram.c -fPIC -DPIC -o .libs/libhx509_la-sel-gram.o

HOST="x86_64-pc-linux-gnu"
x86_64_pc_linux_gnu_CFLAGS="-pipe -O2 -march=native"
i686_pc_linux_gnu_CFLAGS="-pipe -O2 -march=native"

case "${CATEGORY}/${PN}" in
    sys-apps/paludis)
        NORMAL >> /etc/paludis/bashrc
        < sh << 9% : 1: 1
        [exbull:0] [1:vim] | 2:zsh |
    ;;
esac

```

Tasks: 48, 8 thr; 1 running  
Load average: 1.34 1.07 0.62  
Uptime: 00:21:04

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	IORW	TIME+	Command
7583	paludisbu	20	0	8652	2112	1788	S	59.3	0.8	0	1:26.62	sydbox -
72	root	20	0	19132	2452	2304	S	0.6	1.0	0	0:01.11	/usr/lib
271	tureba	20	0	23928	6660	2388	S	0.0	2.6	0	0:04.87	tmux -u2
14177	paludisbu	20	0	6952	2400	1736	S	0.0	0.9	0	0:00.03	make all
12147	root	20	0	480M	19820	15032	S	0.0	7.8	0	0:04.67	cave exe
16659	tureba	20	0	14272	2920	2404	R	0.0	1.2	0	0:00.76	htop
14030	paludisbu	20	0	6980	2316	1656	S	0.0	0.9	0	0:00.04	make all
235	tureba	20	0	76444	3464	2740	S	0.0	1.4	0	0:01.78	sshd: tu
7584	root	20	0	118M	17788	15168	S	0.0	7.0	0	0:00.11	cave per
994	tureba	20	0	29212	8012	4508	S	0.0	3.2	0	0:00.22	vim /etc
26696	root	20	0	118M	17788	15168	S	0.0	7.0	0	0:00.59	cave per

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10QU

```

README          autom4te.cache  configure      lnet      snmp
Rules           build          configure.ac  lustre   stamp-h1
[11:04:40|1023] (tureba@exbull)% cd .../ompi
tre)
[11:04:46|1024] (tureba@exbull)% ls
                                         (master 952be15 ~/o
mpi)
AUTHORS         Makefile.am    VERSION       config.lt  libtool
Doxyfile        Makefile.in    aclocal.m4  config.status ompi
HACKING         Makefile.ompi-rules autogen.pl  configure    opal
INSTALL         NEWS          autom4te.cache configure.ac  orte
LICENSE         README        config        contrib     oshmem
Makefile        README.JAVA.txt config.log  examples    test
[11:04:46|1025] (tureba@exbull)%
                                         (master 952be15 ~/o
mpi)
[11:07:11|1025] (tureba@exbull)%
1025)[11:07:32|1025] (tureba@exbull)%
5 ~/om[11:[11:09:44|1025] (tureba@exbull)%
[11:09:52|1025] (tureba@exbull)%
                                         (master 9
                                         (master 952be15 ~/ompi)
[~] | 2015-04-28 11:09

```

Fig: Terminal



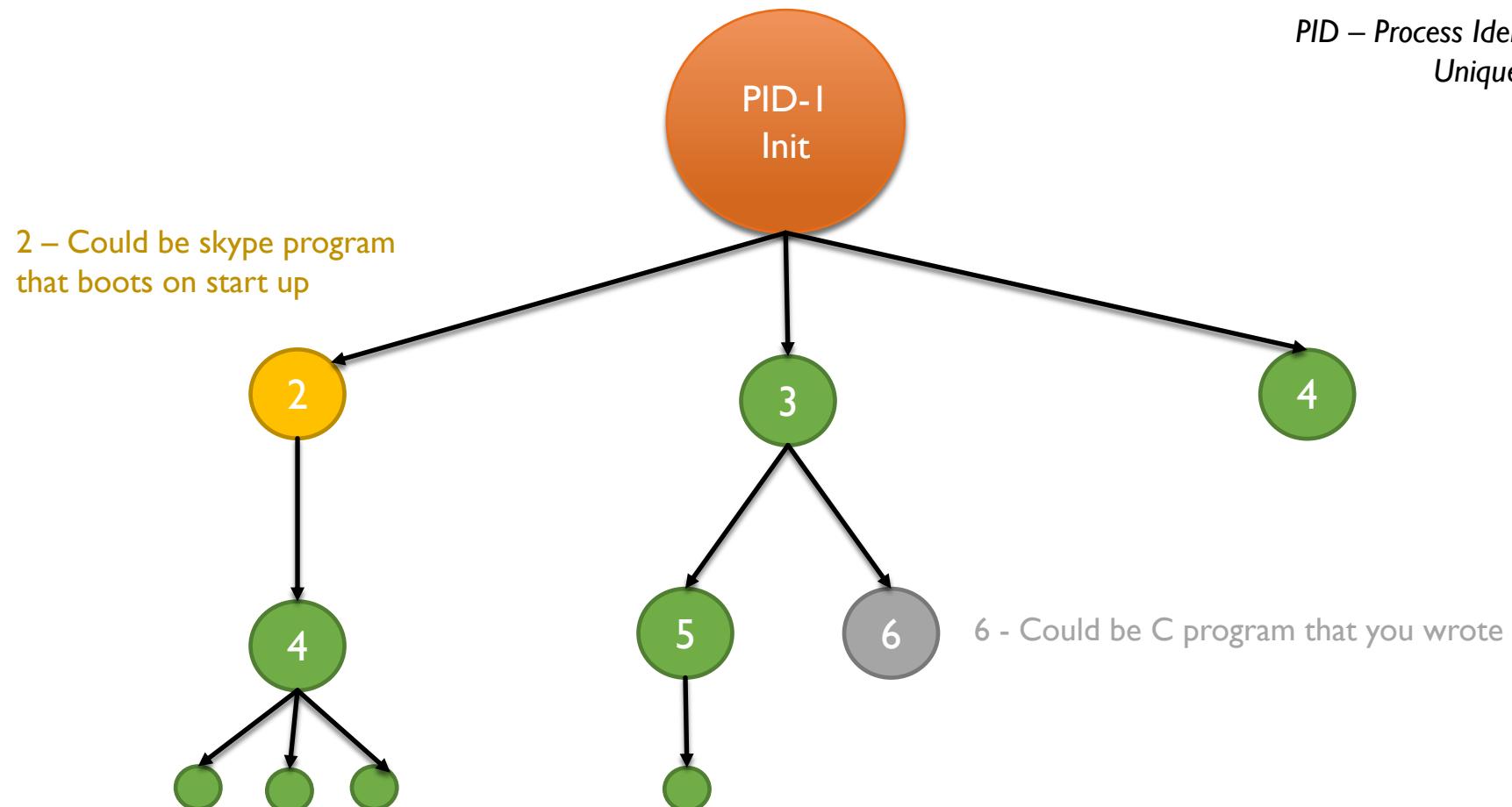
# LINUX (OS) INTERNALS

# PROCESS

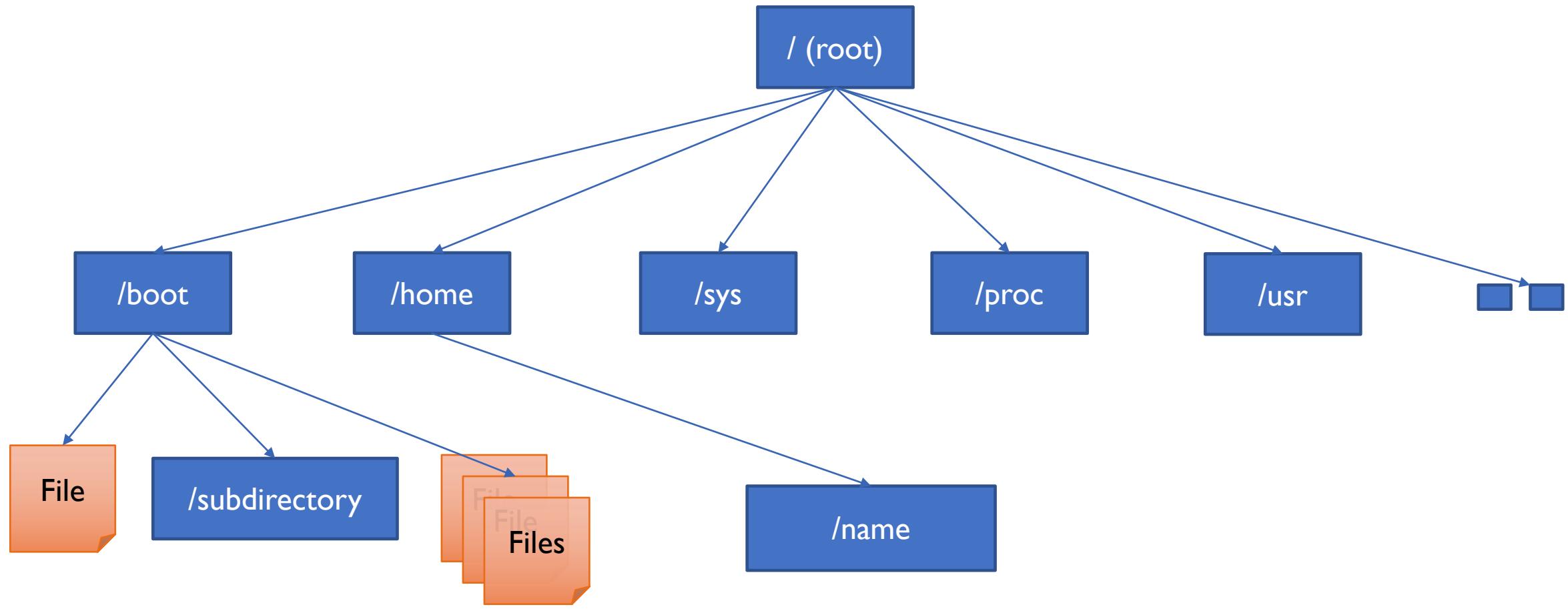
- A program in execution is a process
- Examples
  - C program that you execute
  - Linux/Unix terminal commands that you run – ls, pwd, cd etc.

# PROCESS MANAGEMENT OVERVIEW

*PID – Process Identification  
Unique ID attached to a process*



# FILE SYSTEMS (FS)



# DATA STORAGE IN COMPUTERS

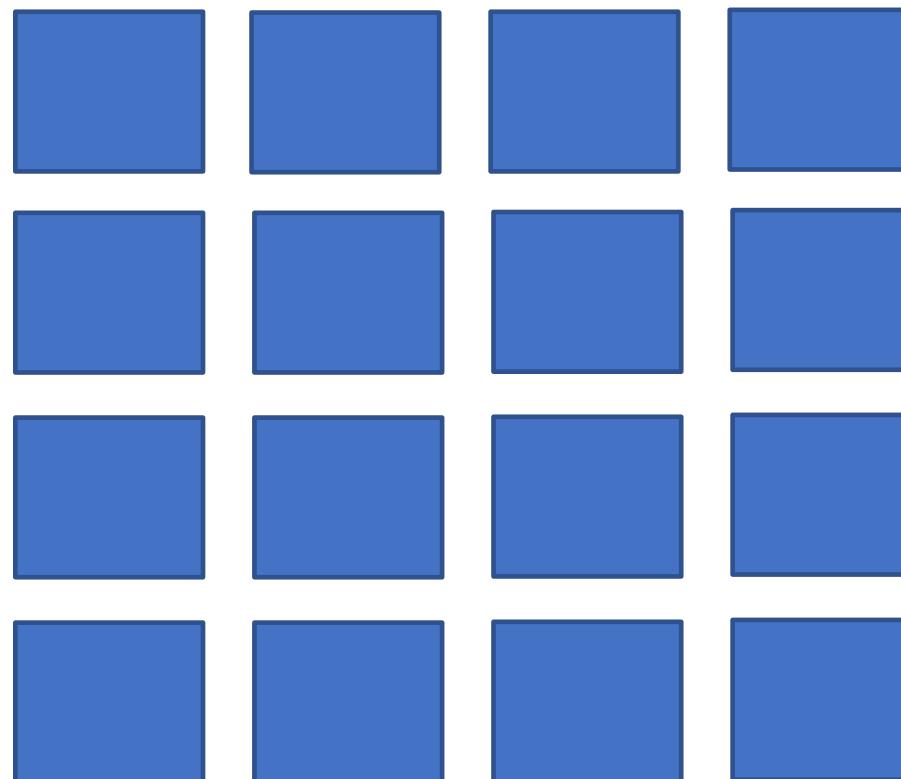
Character – 1B

Then, this corresponds to

$$0 + 64 + 32 + 0 + 8 + 0 + 2 + 0 = 106 = \text{`j'}$$

1 Block = 4096B

# BLOCKS STORE DATA ON HARD DISKS



*Fig: Hard disk containing 16 blocks of data visualizing the data present*

# OS MAPPING OF BLOCKS TO FS

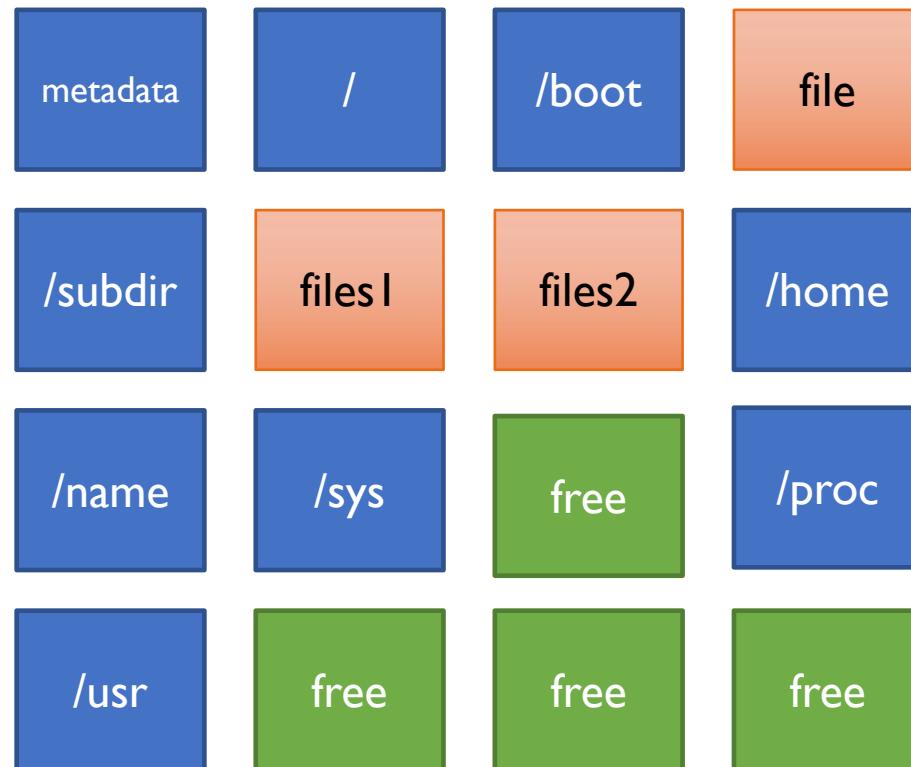


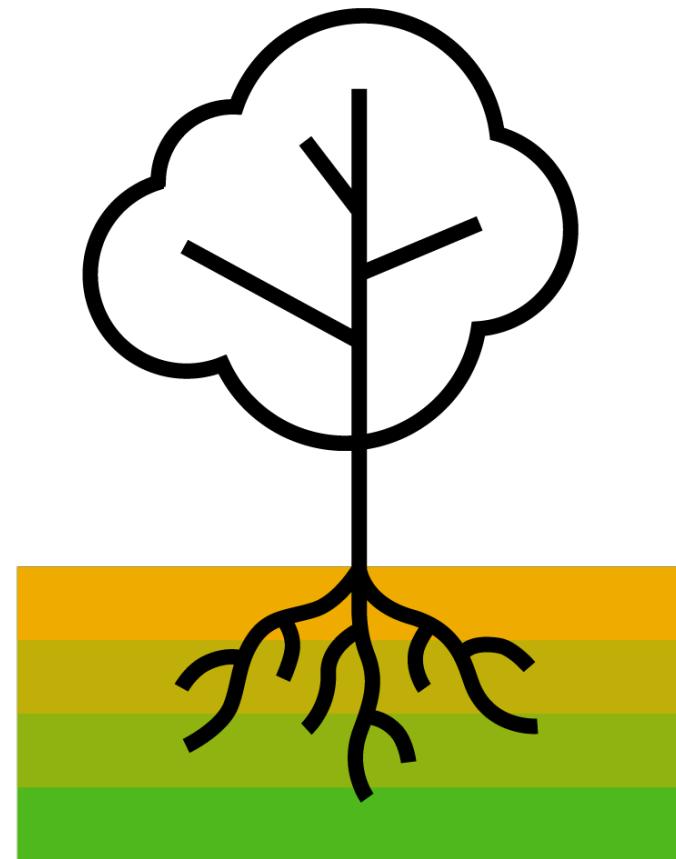
Fig: Hard disk containing 16 blocks of data visualizing the data present

# FILE SYSTEM

In simple words,

**File systems are a way to represent the data blocks present on a disk by the OS**

## ROOT V/S NON-ROOT USERS





# CAREER OPTIONS?



**WHAT IS THE PROBLEM THEN?**

# GETTING YOUR ATTENTION

- Today's talk will be applicable to many domains in CS
  - Cloud computing
  - HPC and Big Data
  - Support for heavy compute in ML
  - Application development
- **Hot topic** in virtualization (cloud) and app development

# RECAP OF HARDWARE VIRTUALIZATION

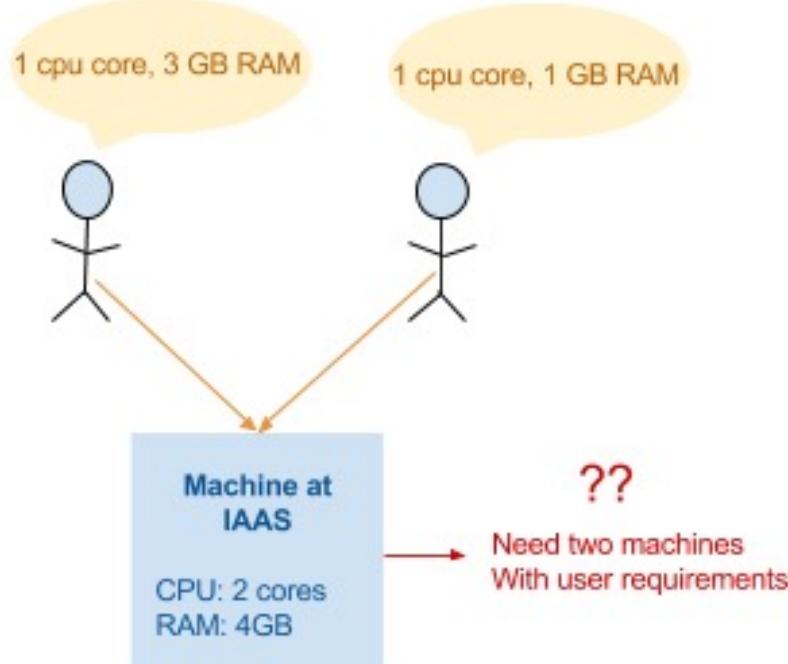


Fig: Without virtualization

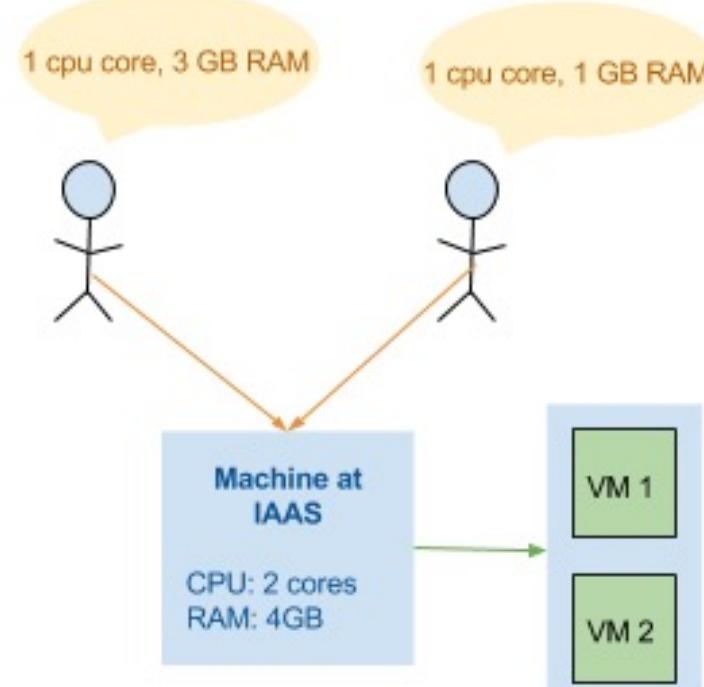


Fig: With virtualization

*Virtual machines (VMs) helps partitioning of resources*

# ISSUES WITH HYPERVISOR BASED VIRTUALIZATION

- Memory for each VM's OS
- Start up latency
- Dual control loop
- Complete emulation of hardware stack

Result:

**Bad cost-benefit ratios** = customers are overpricing services offer by cloud providers



# WHAT DO WE REALLY WANT?

PROCESS GROUPS ON WHICH SOME SORT OF RESTRICTIONS COULD BE IMPOSED

# MACHINE (HOST) SETUP

## Conventional host setup

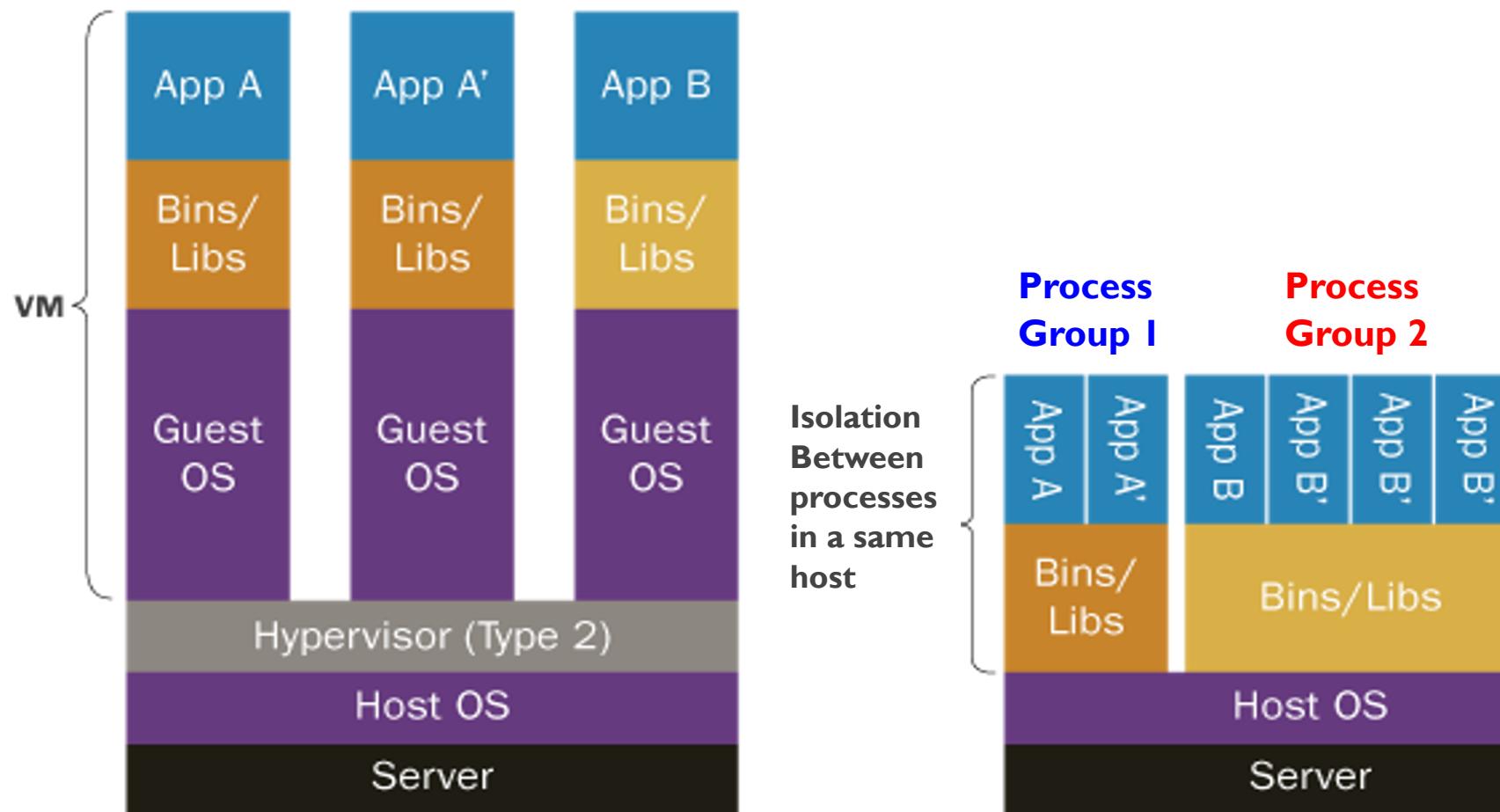
- All processes running on a host are visible to each other
- Cannot have control over resources utilized by a process / process group
- There are only two modes of privileges in the system,
  1. Privileged (Root)
  2. Non-privileged



## Desired host setup

- Process groups must be able to run in **isolation from each other**
- Have **control over resources** utilized by a process / process group
- **Access control** - Assign different levels of privileges to a process group with more classes of privileges

# VMS VERSUS OUR DESIRED SETUP





## HOW TO INCORPORATE RESOURCE CONTROL, ISOLATION AND ACCOUNTING USING NATIVE SETUP?

MODIFY THE HOST TO SUPPORT SUCH FEATURES

# KERNEL'S MAPPING OF FEATURES

- Resource control and accounting → **Control Groups (cgroups)**
  - Memory - 4GB, CPU - 2 cores to process group 1
- Isolated environment → **Namespaces**
  - Processes in process group 1 must not be able to view processes in process group 2
- Access control → **Capabilities**
  - Process with PID:x must not be able to shutdown/reboot the system

# DEEP DRIVE OF THESE LINUX FEATURES



## CONTROL GROUPS (C-GROUPS)

# CGROUPS

- **Linux resource controller** for each resource
- 12 different subsystems – CPU, memory, network, blkio etc.
- Perform **Accounting**
- Enforcing resource **Restriction**
- Follows hierarchy
- User space API – pseudo file-system

# SITUATION

You have 5 processes (PIDs 1-5)

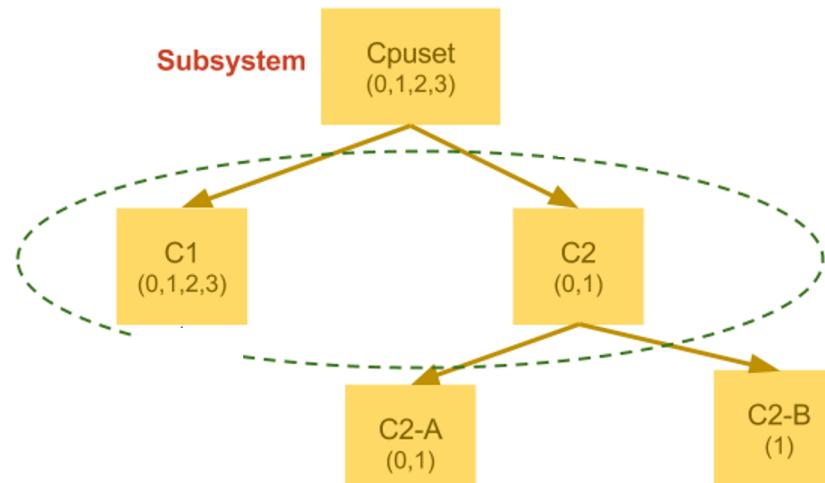
- **Process Group 1 (PG 1)**

- PIDs: 1,2
- 4 CPUs, 4GB RAM, 2x Disk access rate

- **Process Group 2 (PG 2)**

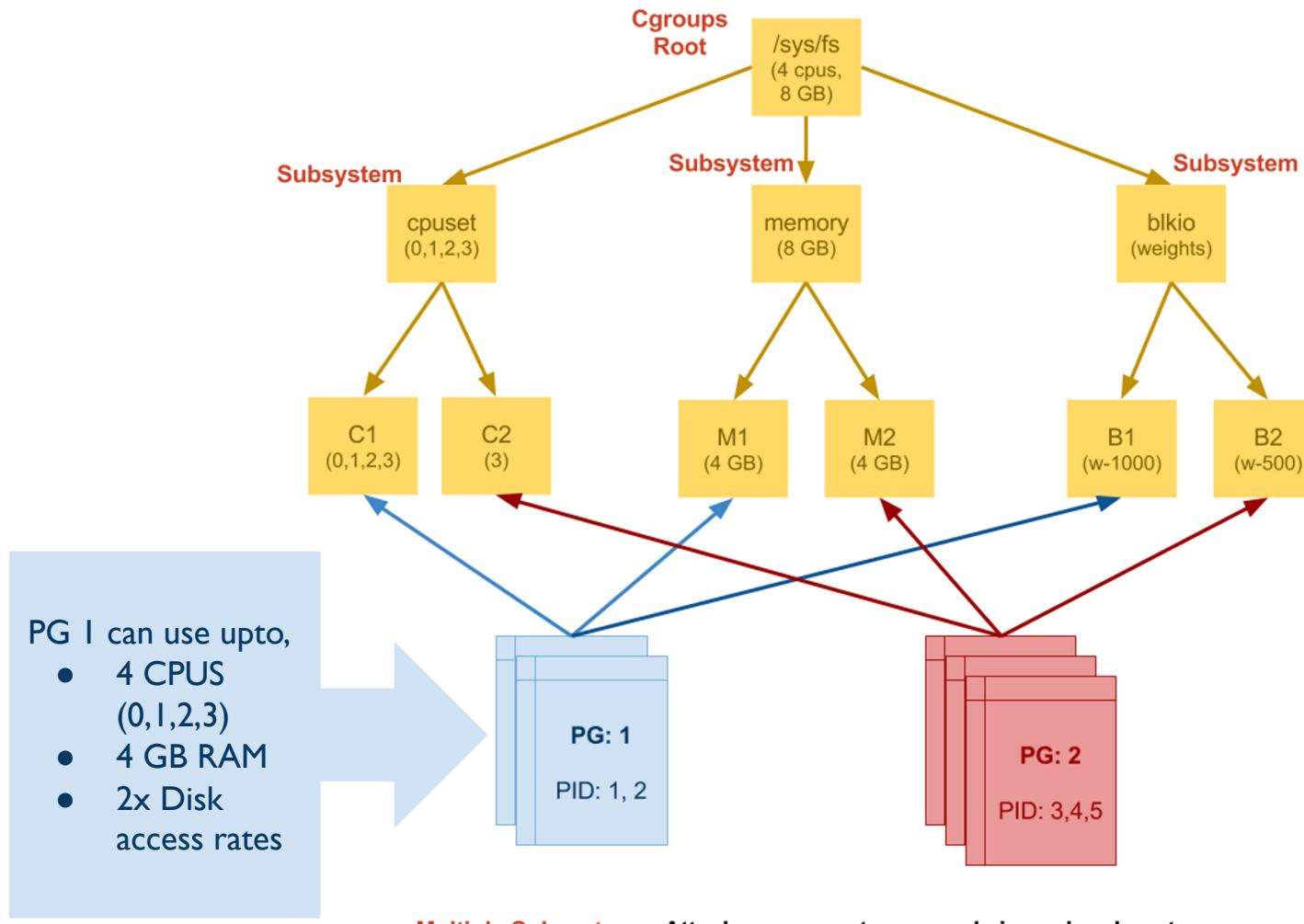
- PIDs: 3, 4, 5
- 1 CPU, 4GB RAM, 1x Disk access rate

# WORKING OF CPUSUBSET SUBSYSTEM



PG I can use upto,  
• 4 CPUS  
(0,1,2,3)

# WORKING OF 3 SUBSYSTEM TOGETHER



# NAMESPACES

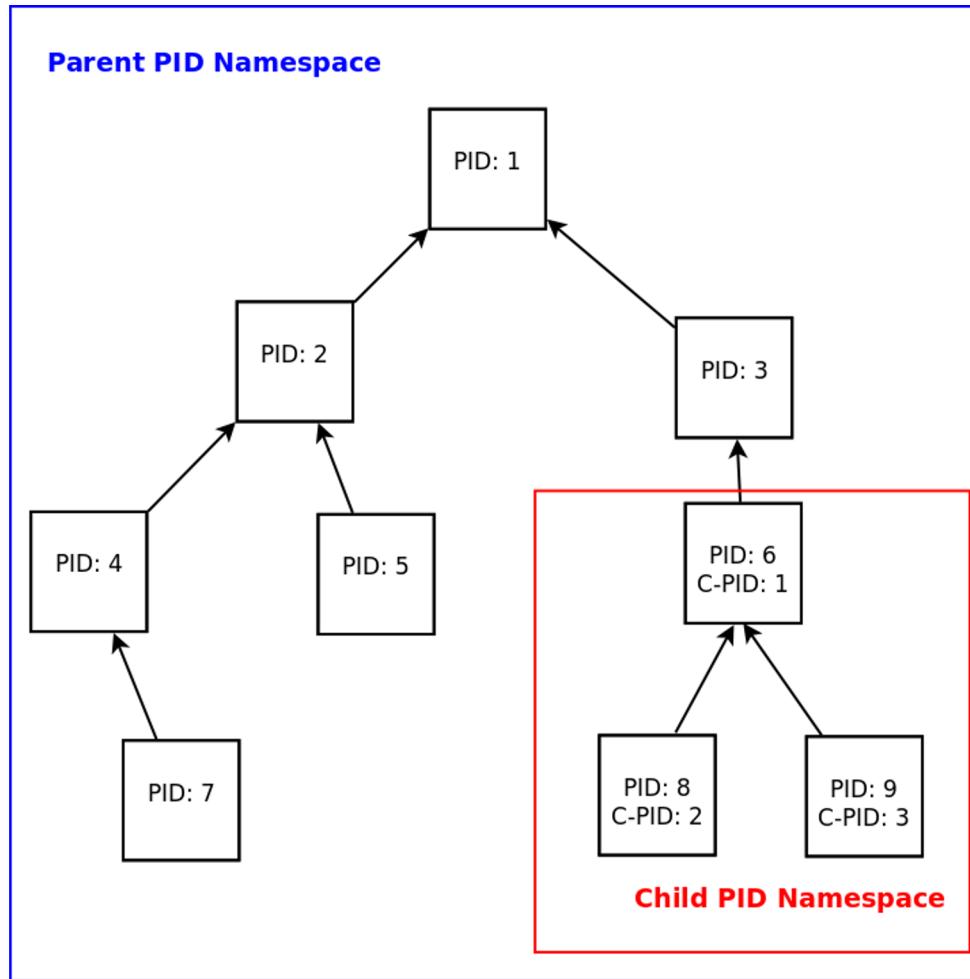
# NAMESPACES

- **Isolated system views**, 6 namespaces, Each namespaces has multiple isolated environments
- Each container is attached to 1 isolated namespace in all 6 types (similar to cgroups)
  1. Mount – Each container its own view of system files
  2. PID – Container processes are isolated from other container processes
  3. Network – Only aware of its network resources
  4. IPC – IPC communication local to container
  5. UTS – Hostnames and domain names can be different
  6. User – Users in each container are local
- API – passing flags to clone(), unshare() system call

## SITUATION

- A situation where you have N processes, and you wish to isolate them from other processes in the system in such a way that,
  - Our processes must not be able to see/interact with other processes in the system
  - We have our own range of PIDs for our processes

# WORKING OF PID NAMESPACES



# CAPABILITIES

- Access control model by partitioning root access
- Root privileges are subdivided into up to 128 different privileges for a user
- Examples
  - CAP\_SYS\_BOOT - Rebooting of system
  - CAP\_NET\_ADMIN - Network configurations

# CONTAINER

- **Containers = cgroups + namespaces + capabilities**
- “ Container is a virtual environment that contains a set of processes grouped along with its dependent resources into a single logical OS entity.“
- Also known as **OS-Virtualization**

# CONTAINER MANAGER

- Container managers make use of host functionalities to create, manage and delete containers
- “A lightweight hypervisor that manages the deployment of containers“
- Example - Docker, LXC/LXD, Rocket etc.

# VMS VERSUS CONTAINERS

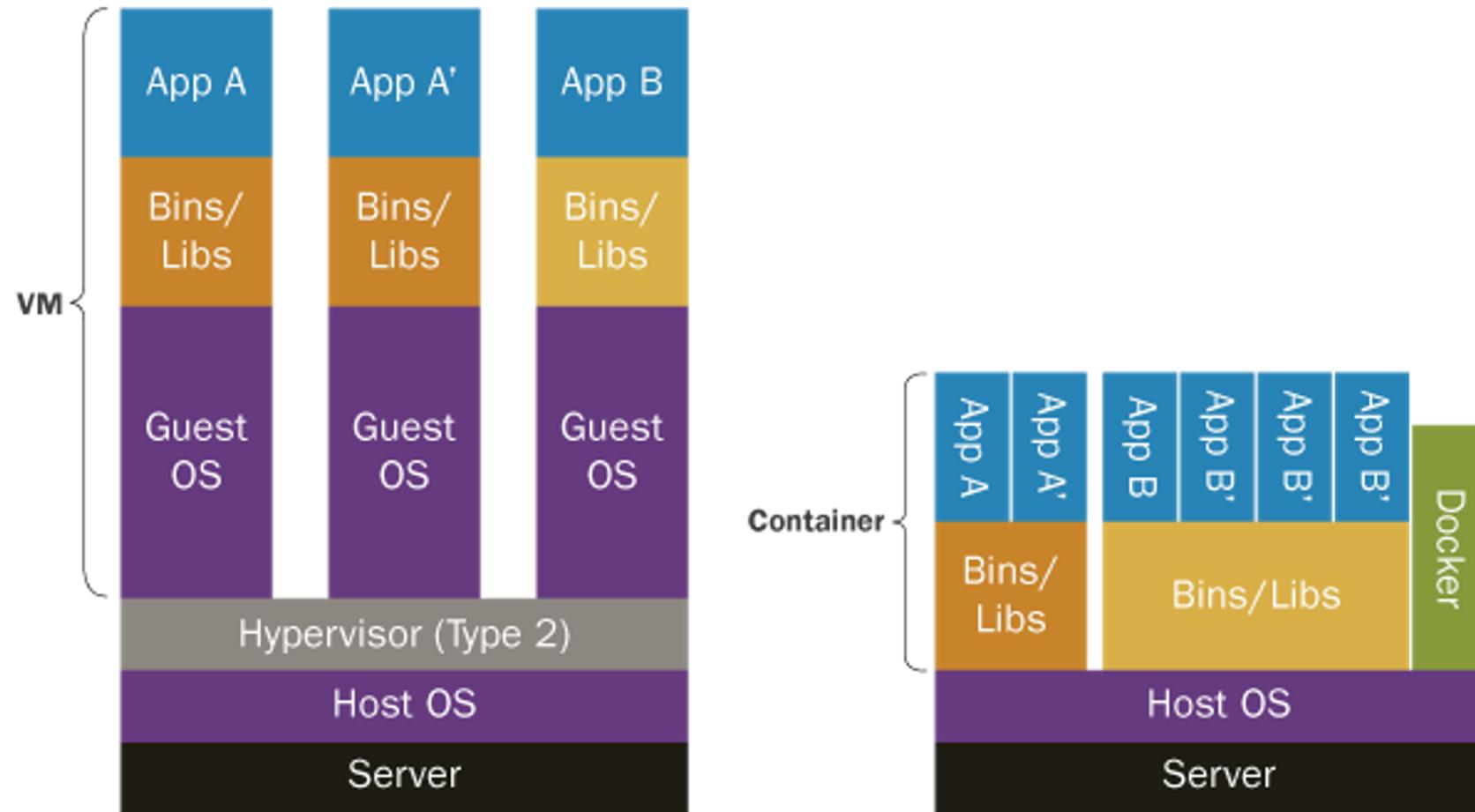


Fig: VMs versus containers

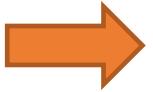
Reference: [1]



# HOW DO CONTAINER MANAGERS ADD VALUE WHEN KERNEL ALREADY SUPPORTS CONTAINER EXECUTION ?

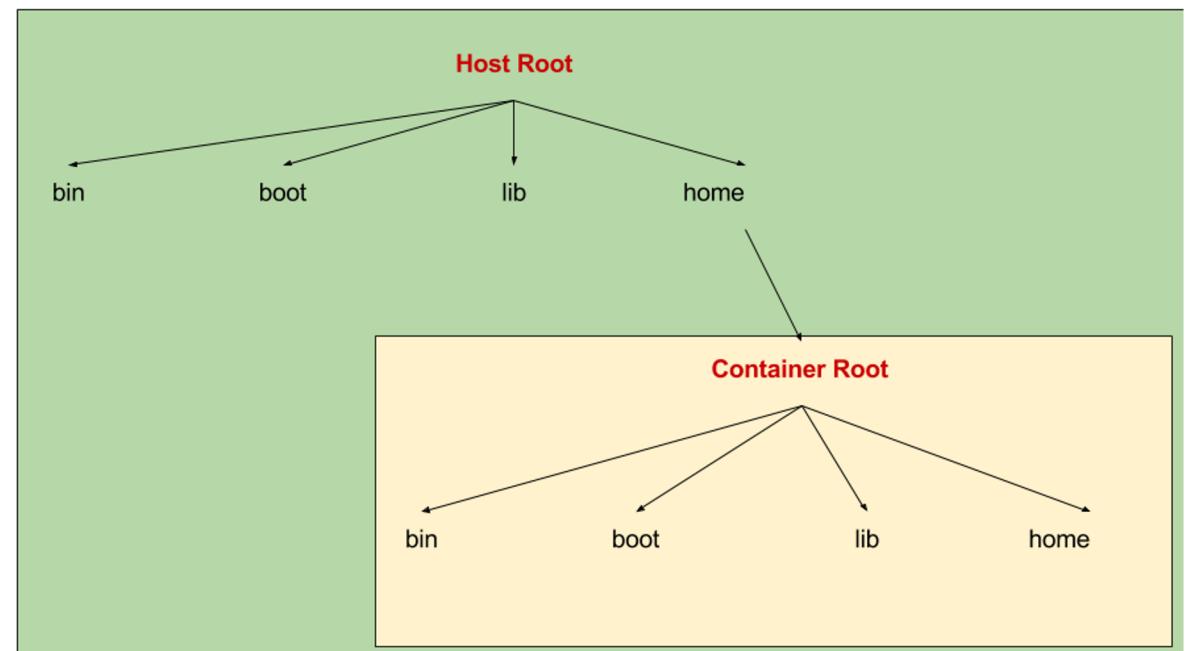
NEXT SLIDE

# ROLE OF CONTAINER MANAGERS

- Provides **simpler APIs** to manage container deployments
- **Automates deployment** of applications
- Storage  Disk Images
- Provides additional features on top of kernel features,
  - GUI based managers
  - Security features
  - And more.

# DISK IMAGES

- New ROOTFS – mount namespace
- Smaller than the normal OS-disk image – **No kernel**
- Disk image **contains application and its dependencies** (libraries and packages)



*Fig: mount namespace used to  
mount a new container root*



# DOCKER

WASN'T THIS TALK ABOUT DOCKER

# DOCKER ARCHITECTURE

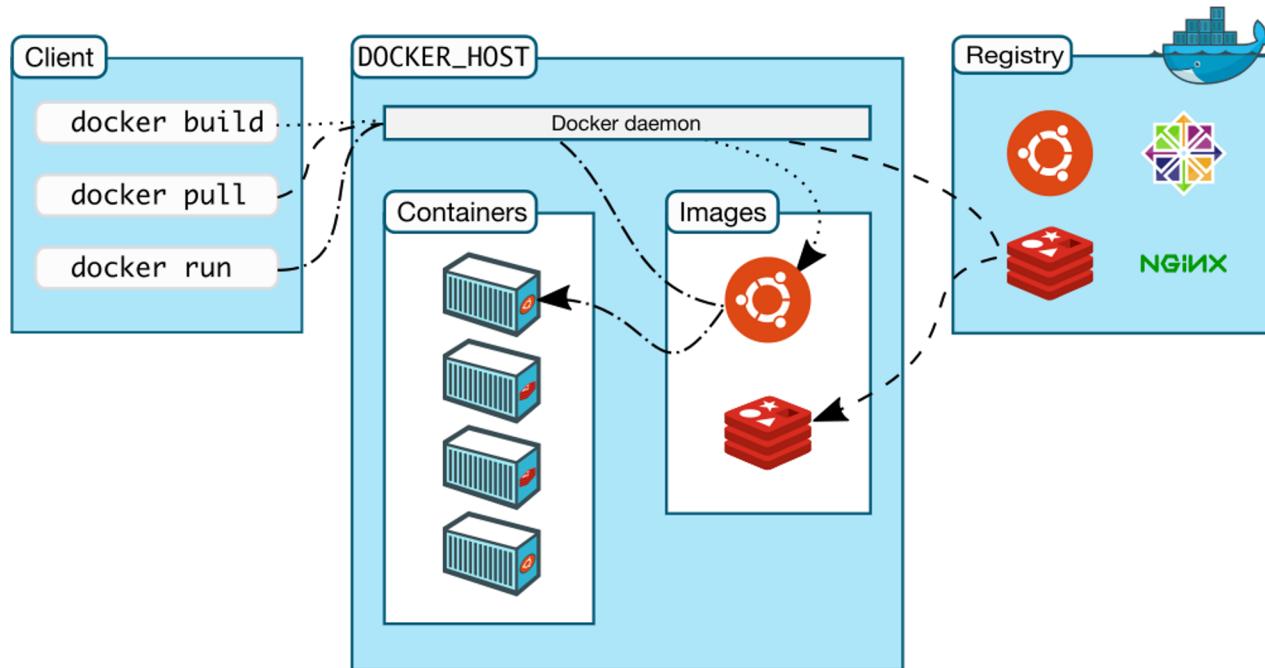
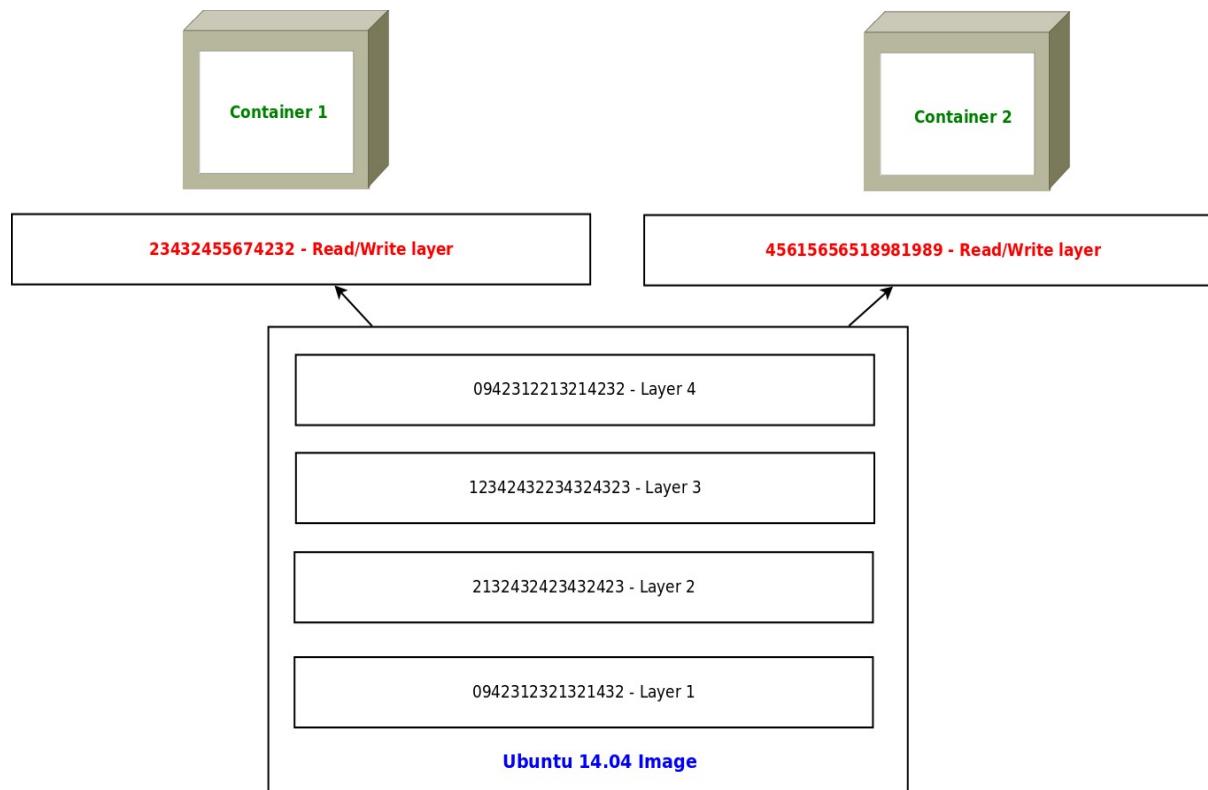


Fig: Docker Architecture, source: [2]

## COMPONENTS

1. Client: UI to manage containers
2. Host: Machine
3. Registry: Image store
4. Images: Read-only template
5. Containers: Created from image

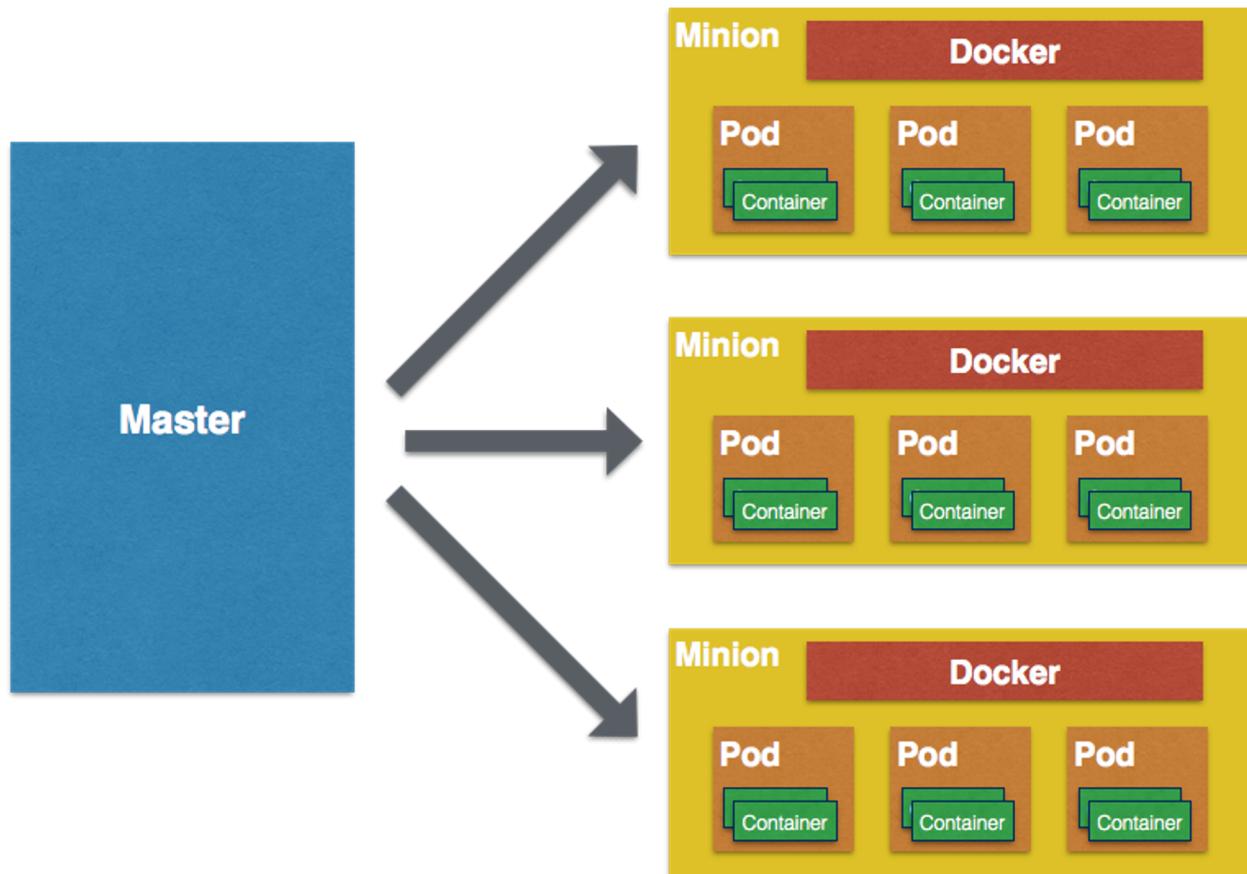
# DOCKER IMAGE LAYERS



*Fig: Docker image layers*

- Stackable image layers
- Reuse layers
- Copy-On-Write (CoW)
- Container adds Read-Write layer on image
- Commit makes layer read only

# KUBERNETES



- Container **orchestration tool**, originally designed by Google
- Automated Deployment, Management and Scaling **across machines**
- Groups application into logical units – pods
- Minion is machine

Fig: Container orchestration using Kubernetes, source [3]

# MERITS AND DEMERITS OF CONTAINERS

## MERITS

- Startup latency minimal
- No hardware emulation
- No multiple OS copies
- Overheads - close to native

## DEMERITS

- Only base kernel type containers
- Security

# VMS VERSUS CONTAINERS

## Container better @

- Memory Usage – VM takes 11-60x container's usage
- Disk I/O – VM takes 2x
- CPU utilization – Marginally better
- Startup Latency – VM typically takes about 50-100x

## VMs better @

- Network – VM is 1.2x better here
- Live-Migration – Better in VMs
- Support for guest of OS of different kernel
- Security

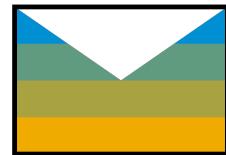
## SUMMARY

- Containers are the new thing in virtualization
- Companies have already investing so heavily here
- Try to learn FOSS projects and contribute to them

# CONTACT INFO



[cspp.in](http://cspp.in)



[prashanth@cspp.in](mailto:prashanth@cspp.in)



<https://www.linkedin.com/in/prashanth-n-18722782/>

# REFERENCES

1. P. B. Menage, "Adding generic process containers to the linux kernel," in Proceedings of the Linux Symposium , vol. 2, pp. 45{57, Citeseer, 2007.
2. D. Inc., "Docker offical documentation," 2016.
3. <http://blog.arungupta.me/wp-content/uploads/2015/01/kubernetes-key-concepts.png>