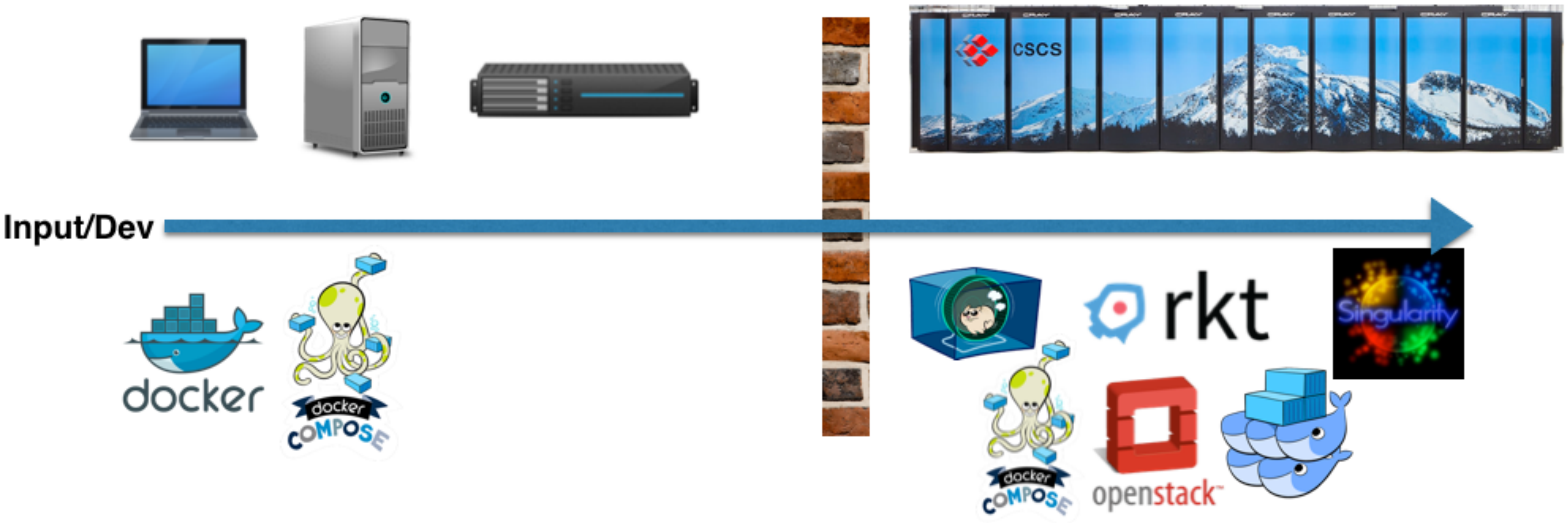
Missing Pieces for HPC

Container Mobility

The Production Wall

Spinning up production-like environment is…

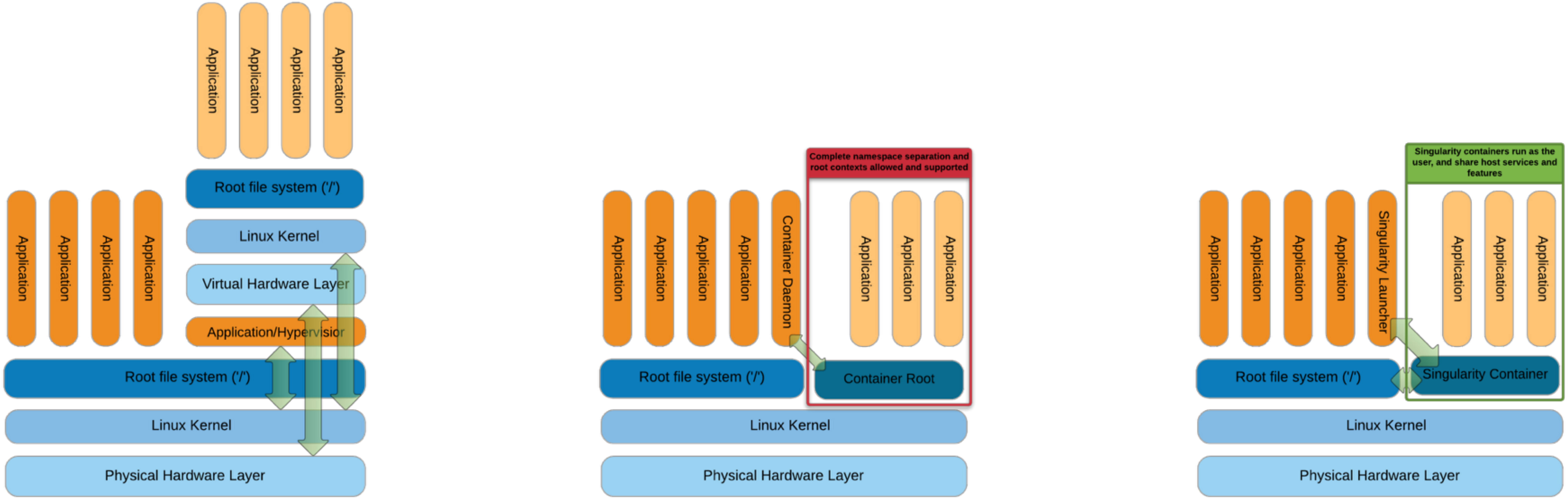
- …not that easy



Hello Singularity

Singularity

- User-land container, leveraging some Namespaces
- creates an executable



MPI Singularity

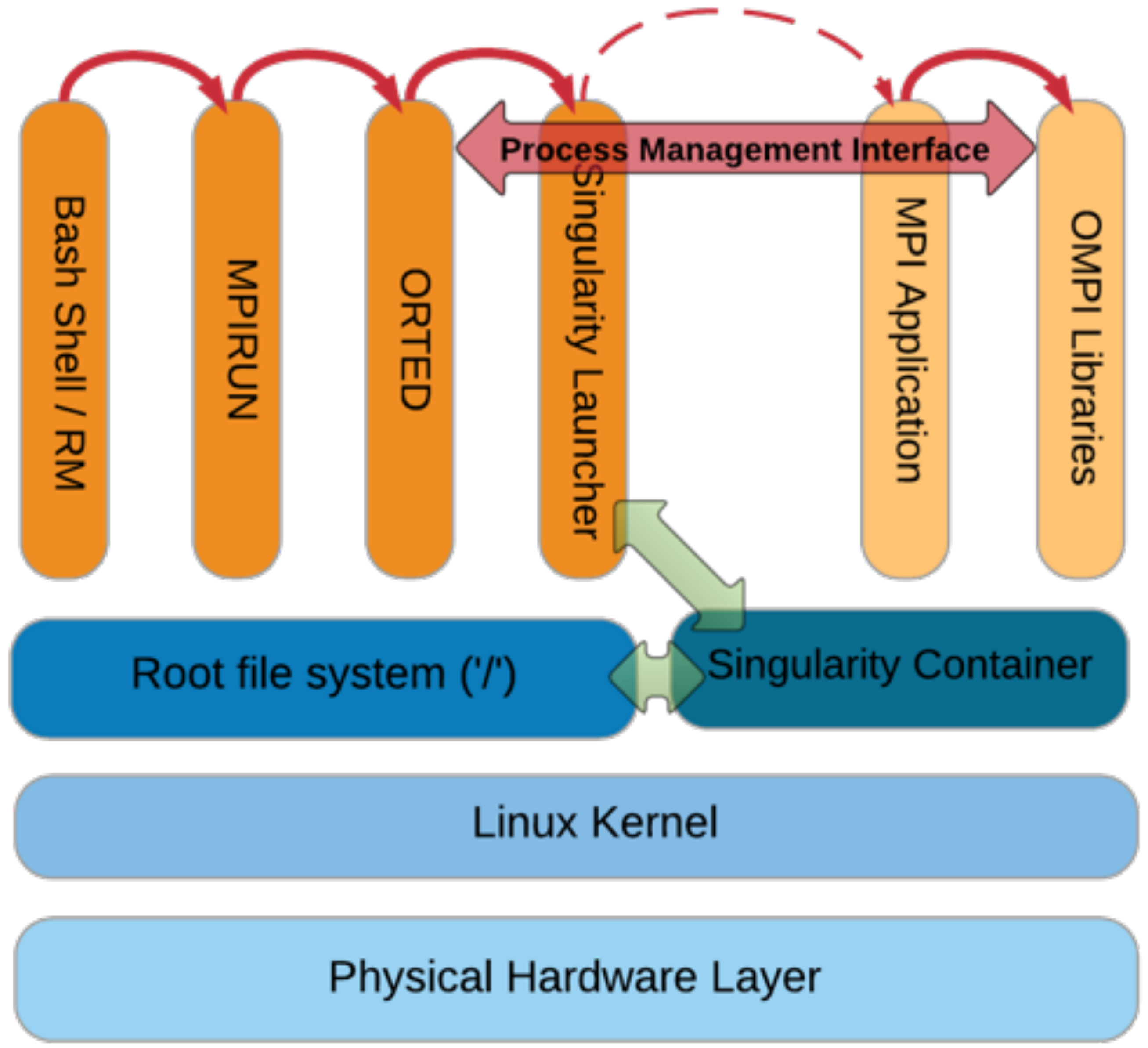
Container Tech creates new NS

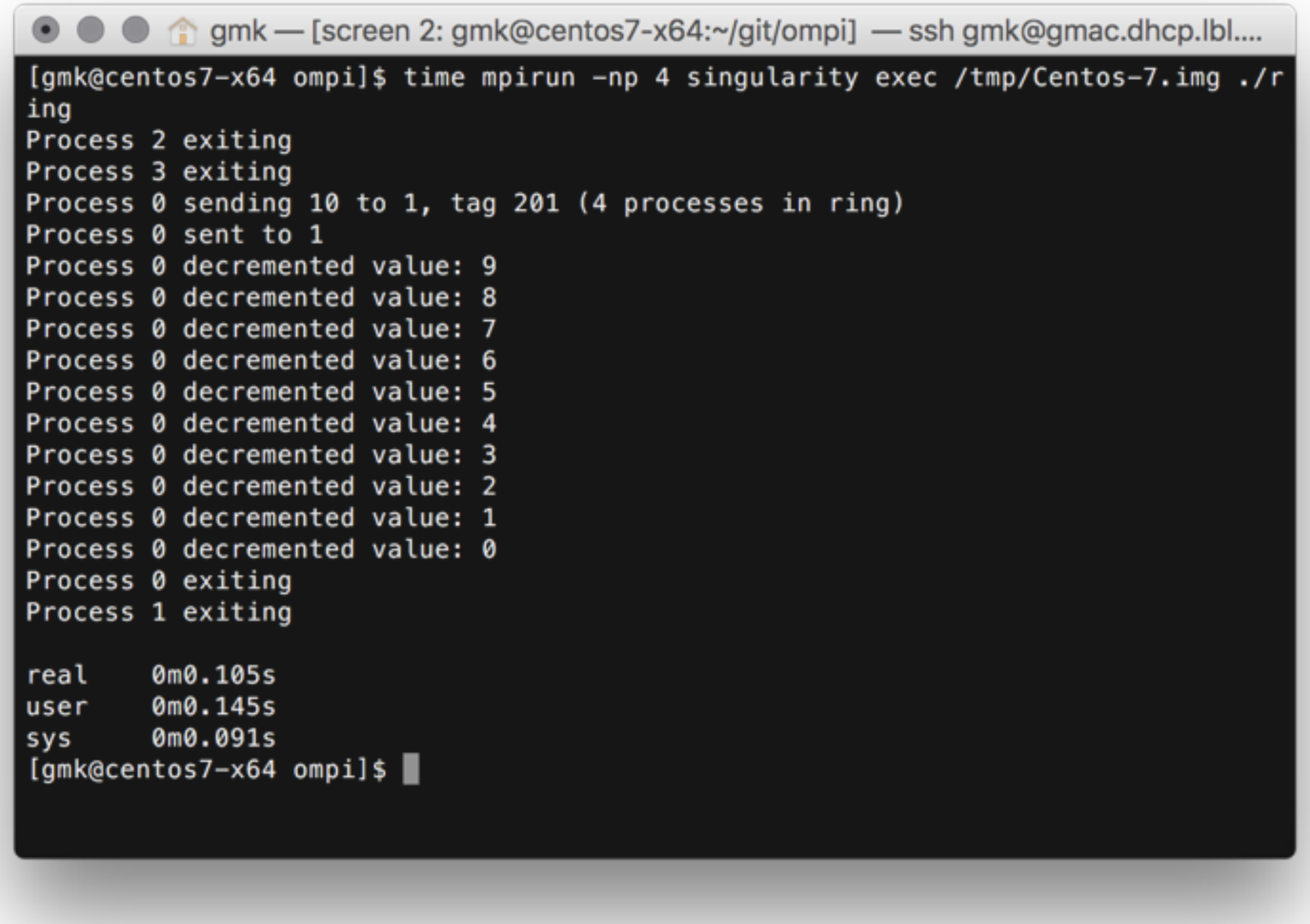
- singularity start from were the user is and trims down from there
- by doing so it put a barrier on-top of what a user can do

MPI Singularity

Singularity is not bound to a daemon w/ API calls

- thus, it integrates (w/ latest) OpenMPI
- comply with the current workflow in HPC w/o changes




gmk — [screen 2: gmk@centos7-x64:~/git/ompi] — ssh gmk@gmac.dhcp.lbl....

```
[gmk@centos7-x64 ompi]$ time mpirun -np 4 singularity exec /tmp/Centos-7.img ./r
ing
Process 2 exiting
Process 3 exiting
Process 0 sending 10 to 1, tag 201 (4 processes in ring)
Process 0 sent to 1
Process 0 decremented value: 9
Process 0 decremented value: 8
Process 0 decremented value: 7
Process 0 decremented value: 6
Process 0 decremented value: 5
Process 0 decremented value: 4
Process 0 decremented value: 3
Process 0 decremented value: 2
Process 0 decremented value: 1
Process 0 decremented value: 0
Process 0 exiting
Process 1 exiting

real    0m0.105s
user    0m0.145s
sys     0m0.091s
[gmk@centos7-x64 ompi]$
```

MPI Singularity

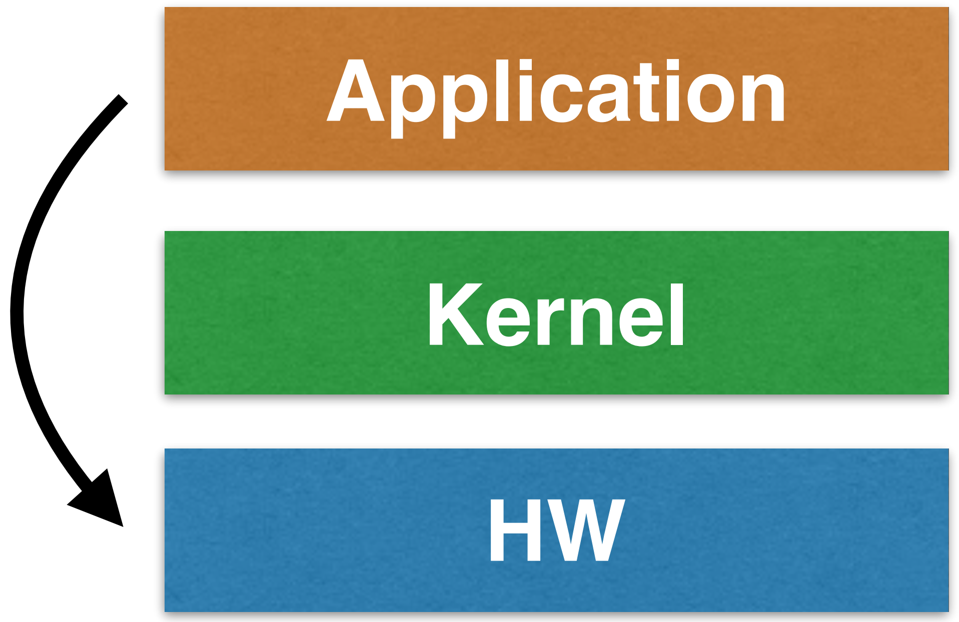
Demo

RDMA Namespace

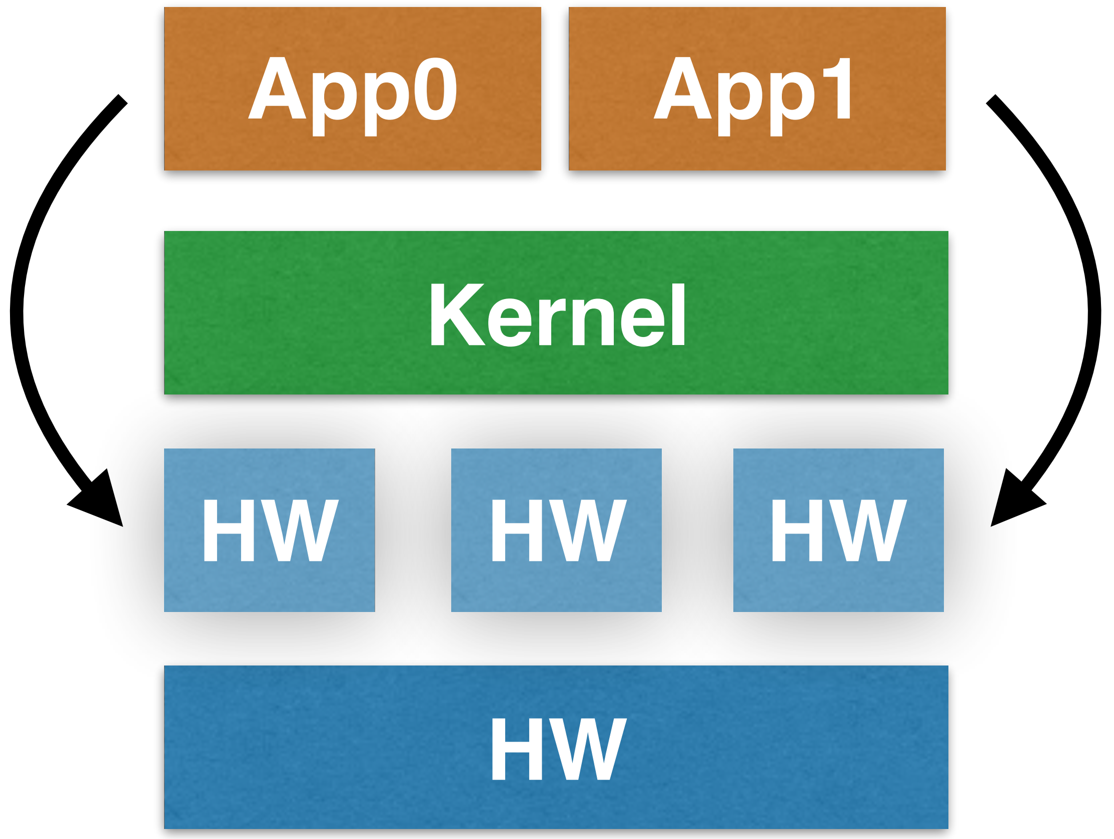
Kernel By-passing

Perfect for single tenancy

- application talks directly to hardware
- nothing in it’s way



1. In VM environment we introduced SR-IOV
   - hardware is exposed as multiple devices within system
   - multiplexing is done in HW
2. Linux containers are extremely volatile
   - if only we could leverage the kernel again,
     as it’s aware of containers



RDMA Namespace

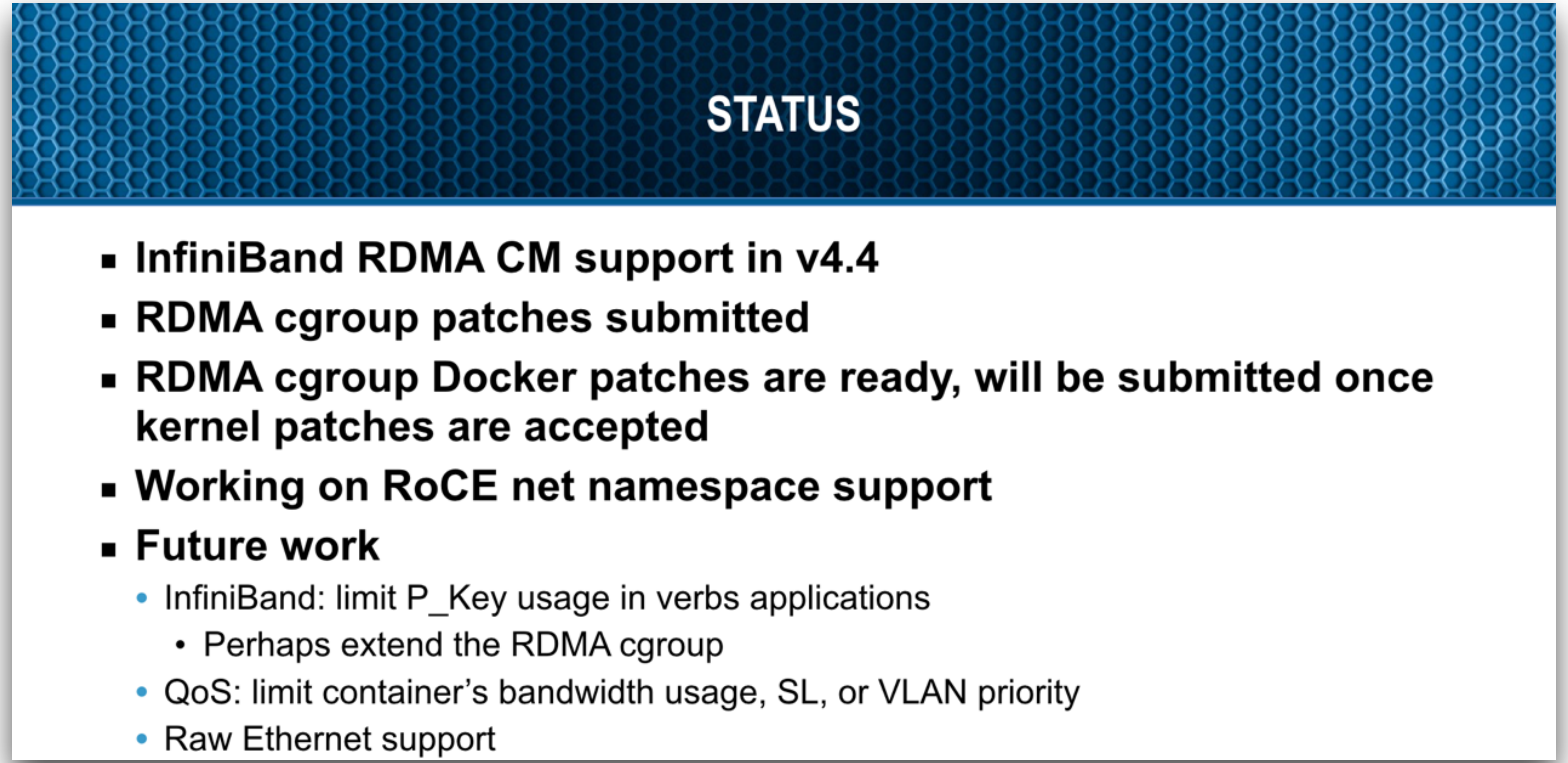
Mellanox pushes code for RDMA

- Namespace
- CGroups

1. By doing so, the Kernel is in charge again

- knows about the resource needs of the all processes

RDMA Namespace #2



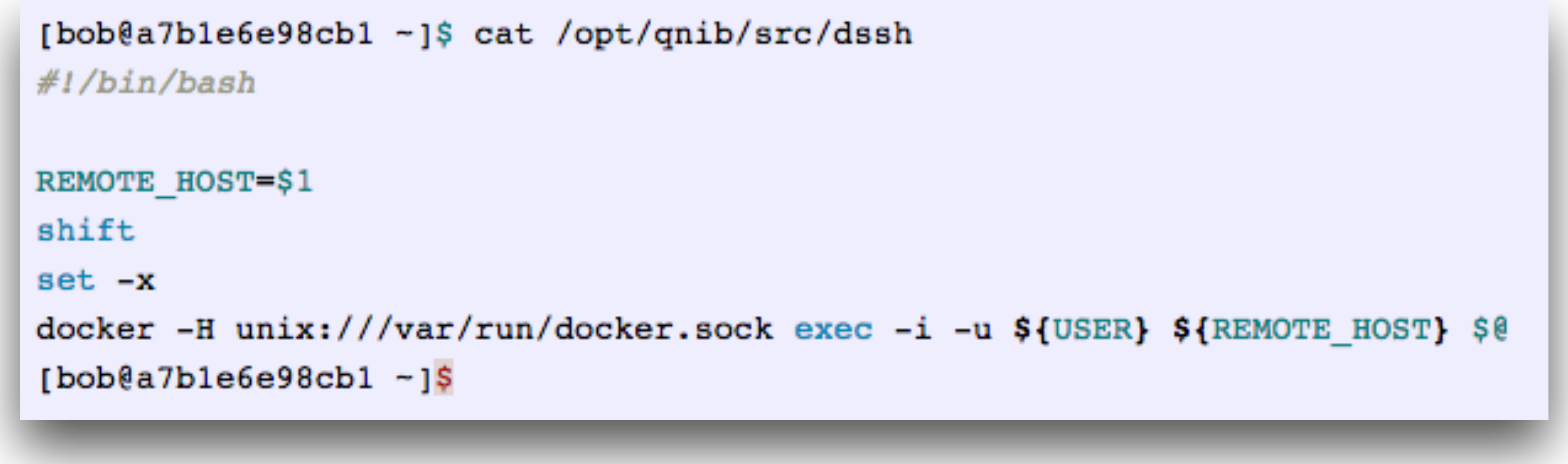
STATUS

- **InfiniBand RDMA CM support in v4.4**
- **RDMA cgroup patches submitted**
- **RDMA cgroup Docker patches are ready, will be submitted once kernel patches are accepted**
- **Working on RoCE net namespace support**
- **Future work**
  - InfiniBand: limit P_Key usage in verbs applications
    - Perhaps extend the RDMA cgroup
  - QoS: limit container’s bandwidth usage, SL, or VLAN priority
  - Raw Ethernet support

SLURM / MPI

MPI Tricks w/ Docker

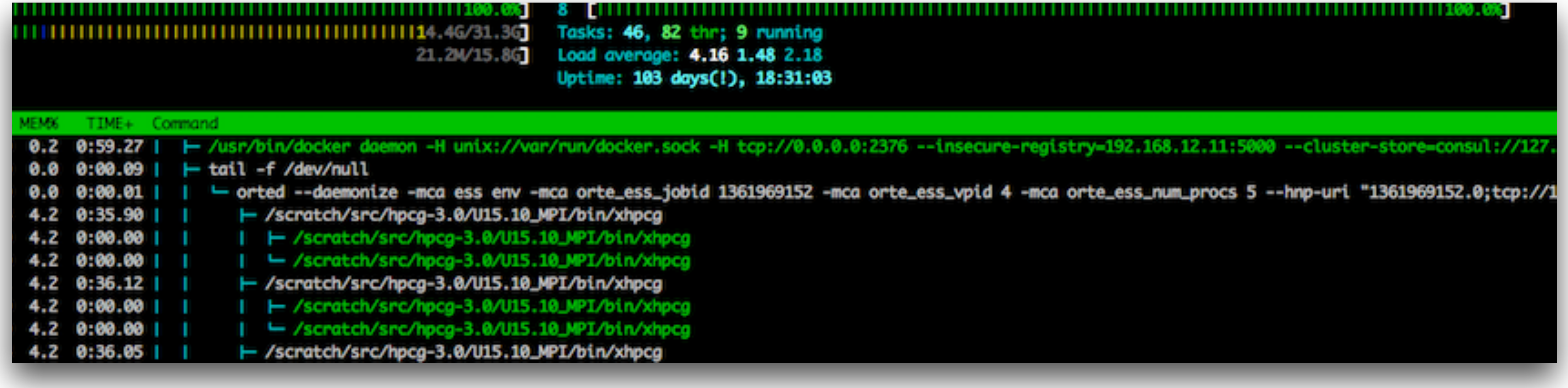
Fake a ssh-client



```
[bob@a7b1e6e98cb1 ~]$ cat /opt/qnib/src/dssh
#!/bin/bash

REMOTE_HOST=$1
shift
set -x
docker -H unix:///var/run/docker.sock exec -i -u ${USER} ${REMOTE_HOST} $@
[bob@a7b1e6e98cb1 ~]$
```

- connects via ***docker exec*** instead of ***ssh***

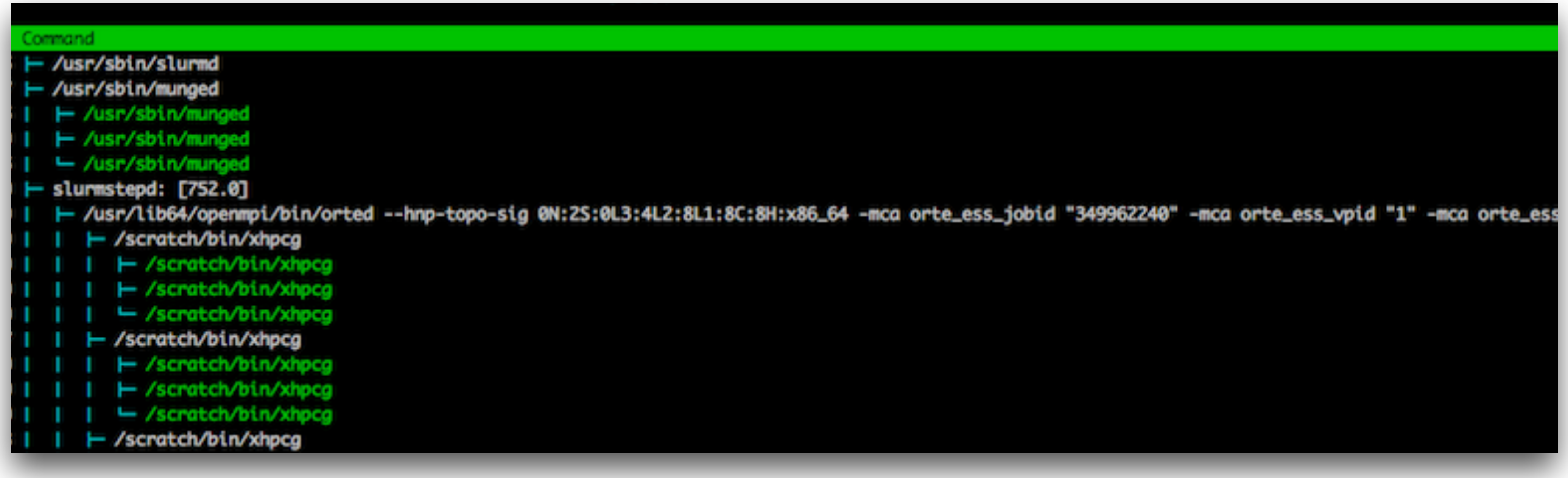


```
||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%]   8  [||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||100.0%]
||||||||||||||||||||||||||||||||||||||||||||||14.4G/31.3G]   Tasks: 46, 82 thr; 9 running
                                              21.2M/15.8G]   Load average: 4.16 1.48 2.18
                                                             Uptime: 103 days(!), 18:31:03

MEM%   TIME+  Command
 0.2  0:59.27 |  ├─ /usr/bin/docker daemon -H unix:///var/run/docker.sock -H tcp://0.0.0.0:2376 --insecure-registry=192.168.12.11:5000 --cluster-store=consul://127.
 0.0  0:00.09 |  ├─ tail -f /dev/null
 0.0  0:00.01 |  |  └─ orted --daemonize -mca ess env -mca orte_ess_jobid 1361969152 -mca orte_ess_vpid 4 -mca orte_ess_num_procs 5 --hnp-uri "1361969152.0;tcp://1
 4.2  0:35.90 |  |     ├─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
 4.2  0:00.00 |  |     |  ├─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
 4.2  0:00.00 |  |     |  └─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
 4.2  0:36.12 |  |     ├─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
 4.2  0:00.00 |  |     |  ├─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
 4.2  0:00.00 |  |     |  └─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
 4.2  0:36.05 |  |     ├─ /scratch/src/hpcg-3.0/U15.10_MPI/bin/xhpcg
```

SLURM needs patching

SLURM uses ***slurmstepd*** to spawn children



```
Command
├ /usr/sbin/slurmd
├ /usr/sbin/munged
│ ├ /usr/sbin/munged
│ ├ /usr/sbin/munged
│ └ /usr/sbin/munged
├ slurmstepd: [752.0]
│ ├ /usr/lib64/openmpi/bin/orted --hnp-topo-sig 0N:2S:0L3:4L2:8L1:8C:8H:x86_64 -mca orte_ess_jobid "349962240" -mca orte_ess_vpid "1" -mca orte_ess
│ │ ├ /scratch/bin/xhpcg
│ │ │ ├ /scratch/bin/xhpcg
│ │ │ ├ /scratch/bin/xhpcg
│ │ │ └ /scratch/bin/xhpcg
│ │ ├ /scratch/bin/xhpcg
│ │ │ ├ /scratch/bin/xhpcg
│ │ │ ├ /scratch/bin/xhpcg
│ │ │ └ /scratch/bin/xhpcg
│ │ ├ /scratch/bin/xhpcg
```

If only

1. Slurmctld should use docker exec instead of fork the process directly (when in docker-mode).
2. MPI could also use docker exec to introduce process on remote system.