Rack Level Scheduling for Containerized Workloads
Qiumin Xu¹, Krishna T. Malladi², Manu Awasthi³
¹USC, ²Samsung, ³IIT-Gandhinagar

Intro to Docker Containers
Docker is an open source software container platform. It is widely adopted
in production environments at Facebook, Uber, Lyft, Ebay etc.

- Each virtualized application includes an entire guest OS (~10s of GB)
- Docker container comprises just
  application and bins/libs
- Shares the kernel with other containers
- Much more portable and efficient

Intro to NVMe and NVMe-over-Fabrics (NVMe)
- NVMe is a storage protocol standard built on top of PCIe [2]
- NVMe SSDs connect through PCIe and support the standard
  • Since 2014 (Intel, Samsung)
  • Enterprise and consumer variant
- Samsung XS1715 Enterprise Level NVMe SSD:
  • 4K random reads/writes: 750K/15K IOPS
  • 128K sequential reads/writes: 3/1.4 GB/s

- NVMe is a new remote storage technique which allows the high performance
  NVMe interface to be connected to RDMA-capable networks [1].
- Advantages of NVMe
  • Retains NVMe performance over network fabrics
  • Eliminates unnecessary protocol translations
  • Enables low latency high IOPS remote storage

Co-located Containerized Services Suffer from Long Tail Latency
- Random write-access latency distribution of Samsung PM1725
  NVMe SSD running a datacenter-level scheduler
- 80% of the storage access latency is within 100us, while 1% of the requests have latency longer than 3.9 ms.
- Normalized tail latency when running concurrent Cassandra services (results are normalized to the p99 latency of workload A)
- Tail latency increases exponentially as more services are concurrently executing on the same server

Rack Level Scheduling
Another interesting observation is that of the eight workloads, the worst
case p99 latency is much higher than the best p99 latency. This indicates that
shifting the worst performing job to another node with more resources
will result in decreasing the large gap between best and worst case
latencies. Traditionally, this has been very difficult since migrating jobs
between machines also meant migrating the data associated with those
jobs. However, with techniques like NVMe becoming increasingly common
within a rack, this hurdle is becoming easier to overcome.

- Rack Level Scheduling (RLS): p99 latency reduced by 3.8x and p99 latency was reduced by 3.6x.

Conclusion
With the advent of high performance remote storage techniques such as
NVMe, workload migration between servers on the same rack becomes
much cheaper. We propose rack level scheduling that exploits fast, remote
storage, to provide 3.8 reduction in tail latency.

Results
Experiment Setup:
1) 4x dual-socket Intel Xeon ES, 48HT CPU
2) One client driving YCSB benchmark, two Cassandra server nodes, one
remote storage server
3) NVMe over 40Gbps RDMA capable Ethernet using RoCE protocol

- Packing: 8 Cassandra containers are packed and concurrently run on
  one server
- Mesos: We set up mesos + marathon + zookeeper frameworks to
  allocate Cassandra containers.
- Rack Level Scheduler (RLS): p99 latency reduced by 3.8x and p99
  latency was reduced by 3.6x.

References
[1] Q. Xu, et al., Performance analysis of containerized applications on local
and remote storage, in Proc. of MSST 2017
[2] NVMe Express, "NVMe Express = scalable, efficient, and industry standard,"
www.nvmexpress.org, 2016

Acknowledgements and Contacts
• This work was done at Memory Solution Lab, Samsung Semiconductor Inc.
• Please feel free to contact qumin@usc.edu for any questions