Narrowing the Gap: Effects of Latency with Docker in IP Networks

Corbin Higgs – Student (corbinhiggs@clemson.edu)

Jason Anderson – Advisor (jwa2@clemson.edu)

Motivation

Recent work has shown that lightweight virtualization such as Docker containers can be used in HPC to package applications with their runtime environments [1]. In many respects, applications in containers perform similarly to native applications [2, 3]. Other work has shown that placing an application in a Docker container has adverse effects to the latency variation of communications with that application [4]. This latency variation may have an impact on the performance of some HPC workloads, especially those dependent on synchronization between processes [5].

Research Objectives

In this work, we measure the latency characteristics of messages to and from Docker containers, and then compare those measurements to the performance of real-world applications. Our specific goals are to:

- Measure the changes in mean and variation of latency with Docker containers
- Study how this affects the synchronization time of MPI processes
- Measure the impact these factors have on real-world applications such as the NAS Parallel Benchmark (NPB)

Methodology

Typical Docker applications use the Linux bridge or Open vSwitch to direct traffic to and from applications in containers. To understand how applications are affected by both the software bridge and the container itself, we established four environments to conduct each benchmark:

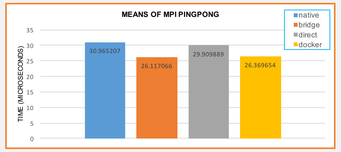
- **native** – native application with normal access to the network
- **bridge** – native application using a veth interface attached to a Linux bridge
- **direct** – Docker application with the network interface directly assigned to the container
- **docker** – Docker application using a veth interface attached to a Linux bridge

Software used:
- Ubuntu 14.04 LTS, kernel version 3.13.0-68-generic.
- Docker 1.11.2
- OpenMPI 1.6.5
- NAS Parallel Benchmark for MPI 3.3

Hardware used:
- Cloudlab [6] machines at the Wisconsin site:
- Cisco C220 M4 rack server with
  - two Intel ES-2660 v3 10-core CPUs at 2.60 GHz
  - 160GB ECC memory
  - dual-port Intel X520 10Gb NIC
- Cisco Nexus C3172PQ top-of-rack switch connecting all test machines.

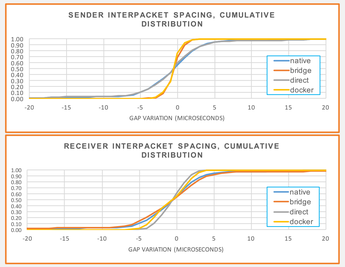
Microbenchmarks

We measured the mean latency imposed by the test environments by conducting a ping-pong test between two MPI processes on separate nodes. We placed the receive side of the test in the test environment, while the sender ran natively to avoid doubling the effect. In each test, the time required for 1000 iterations was measured, and then the average result is reported.

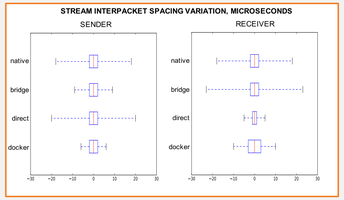


Overall, **native** had the highest mean of every data set, followed closely by **direct**. Surprisingly, tests using extra software layers of the Linux bridge and veth interface (**bridge** and **docker**) had lower means than native interface access.

To measure the latency variation, we send a constant rate stream of messages from an MPI process on one node to an MPI process on another node. To help determine where variation occurs, we further divide the tests into send-side and receive-side variation. We then calculate and report this inter-message spacing.

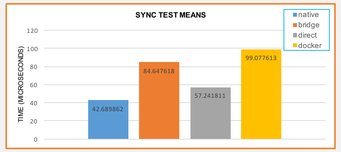


On the sending side, using the Linux bridge seemed to decrease latency variance. On the receiving side, however, the opposite seems to be true; the Linux bridge increased the latency variance. However, placing the receiver in a Docker container decreased the variance in both network configurations.



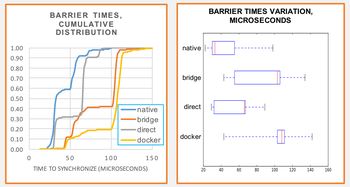
Synchronization

For these tests, we measured the time required for 8 MPI processes on separate nodes to synchronize to a MPI_Barrier() call. Unlike the microbenchmarks, all nodes were under the testing environment. We report the mean and distribution of the times below.



Unlike the microbenchmarks, **native** and **direct** had lower means and deviations, while measurements in environments using the Linux bridge were more variable. Each environment had an unexpected bimodal plateau at similar positions, with between twenty-five and seventy-five percent of the data in one peak.

We also observe that the environments using a container result in a higher mean synchronization time, despite the lower receive-side latency variation and equivalent mean and send-side latency variation of the microbenchmarks.

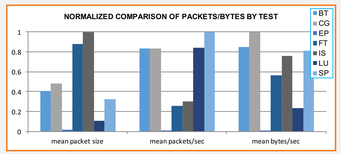


Application Benchmarks

We ran the NAS Parallel Benchmark with seven of the nine benchmarks compiled with the problem sizes below. Problem sizes were chosen to make each test run at least 2 seconds to ensure an accurate sample. First, we measured the total number of packets that were transmitted between nodes for each program, then ran 30 iterations of each benchmark on eight nodes with the appropriate maximum number of cores.

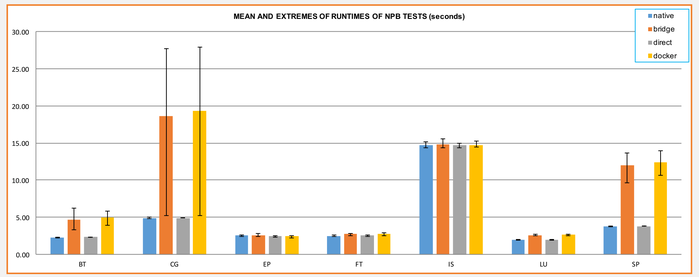
Name	Size	Procs	Packets (M)	GB
Block Tri-diagonal solver (BT)	A	144	5.62	13.50
Conjugate Gradient (CG)	B	128	11.98	33.98
Embarrassingly Parallel (EP)	C	128	0.01	0.00
3D fast Fourier Transform (FT)	B	128	1.88	9.80
Integer sort (IS)	D	128	13.01	77.08
Lower-Upper Gauss-Seidel Solver (LU)	A	128	4.80	3.10
Scalar Penta-diagonal Solver (SP)	A	144	11.06	21.20

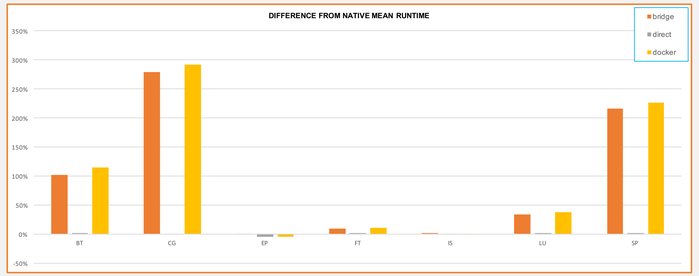
To estimate how dependent each benchmark is on communications, we compare each benchmark's mean packet size and transmission rate as compared to native run times.



To estimate the impact of each environment on runtime, we compare each test's min, max, mean, and difference from native mean.

As with the synchronization tests, **bridge** and **docker** added a significant amount to the mean and variation on BT, CG, LU, and SP. We observe that tests with lower packet transmission rates are less affected by the bridge and veth interface.





Discussion

Although we observed in the microbenchmarks that the Linux bridge and veth interface lowered the mean and sender-side variation of message latency between MPI applications, we saw that synchronization time increased for the same test environments. Furthermore, we see that application benchmarks that send many packets per second are more affected by the extra software switch than those with direct access to the network interface.

We hypothesize that a buffer in the bridge or veth software layers may be small enough to cause packets to be dropped, causing TCP resends to occur and lower the overall performance of applications that send bursts of packets. This would explain the performance drops for MPI_Barrier and the NPB benchmarks BT, CG, LU, and SP when the Linux bridge is used.

We also note that in all cases, the Docker container seemed to have little effect on application performance when direct access to the network interface is available, despite the observed higher mean synchronization time of MPI_Barrier. We actually saw a 4% lower runtime for the Embarrassingly Parallel benchmark using Docker with both types of network interfaces.

Future Work

In our continuing work, we are investigating how buffer sizes and packet drops in the Linux bridge and veth interface play a part in performance of bursty application traffic. We are also interested in how alternative network interface technologies such as InfiniBand and SR-IOV can be used for practical direct interface assignment to Docker containers.

References

[1] Higgins, Joshua, Violeta Holmes, and Colin Venters. "Orchestrating Docker Containers in the HPC Environment." In International Conference on High Performance Computing, pp. 506-513. Springer International Publishing, 2015.

[2] Xavier, Miguel G., Marcelo V. Neves, Fabio D. Rossi, Tiago C. Ferreto, Timoteo Lange, and Cesar AF De Rose. "Performance evaluation of container-based virtualization for high performance computing environments." In 2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pp. 233-240. IEEE, 2013.

[3] Ruiz, Cristian, Emmanuel Jeanvoine, and Lucas Nussbaum. "Performance evaluation of containers for HPC." In European Conference on Parallel Processing, pp. 813-824. Springer International Publishing, 2015.

[4] Anderson, Jason, Hongxin Hu, Udit Agarwal, Craig Lowery, Hongda Li, and Amy Apon. "Performance considerations of network functions virtualization using containers." In 2016 International Conference on Computing, Networking and Communications (ICNC), pp. 1-7. IEEE, 2016.

[5] Martin, Richard P., Amin M. Vahdat, David E. Culler, and Thomas E. Anderson. Effects of communication latency, overhead, and bandwidth in a cluster architecture. Vol. 25, no. 2. ACM, 1997.

[6] Cloudlab. http://cloudlab.us/.