Avaliação do Docker Volume e do NFS no Compartilhamento de Sistemas de Arquivos em Contêineres

Marco A. T. Schaefer[1], **Naylor G. Bachiega**[1], **Paulo S. L. Souza**[1], **Sarita M. Bruschi**[1]

[1]Universidade de São Paulo - ICMC-USP
Av. Trabalhador São-carlense, 400 – São Carlos – SP – Brasil

{marco.schaefer,naylor}@usp.br, {pssouza,sarita}@icmc.usp.br

Abstract. *The current demand for computing power requires the use of new resources in order to increase energy efficiency, reduce operating costs, reduce data center space, and thereby reduce carbon emissions into the atmosphere. In this context, containers represent a virtualization technology that attempts to address these demands. Therefore, evaluating container performance is critical to large-scale adoption. This paper investigates, through an experiment, the container response time for Docker Volume and NFS, as there is no such analysis in the related works. The results show that there is a significant difference between all tests with Docker Volume and asynchronous NFS and some significant differences with synchronous NFS.*

Resumo. *A atual demanda por poder computacional exige que novos recursos sejam utilizados para aumentar a eficiência energética, reduzir custos operacionais, reduzir espaço nos centros de dados e, consequentemente, diminuir a emissão de carbono na atmosfera. Neste contexto, contêineres representam uma tecnologia de virtualização que tentam resolver essas demandas. Dessa forma, avaliar seu desempenho é fundamental para sua adoção em larga escala. Este artigo investiga, através de um experimento, o desempenho em termos de tempo de resposta de volumes para contêineres Docker Volume e NFS, visto que tal análise não foi encontrada nos trabalhos relacionados. Os resultados mostram que há diferença significativa entre todos os testes com o Docker Volume e o NFS assíncrono e alguns diferenças significativas com o NFS síncrono.*

1. Introdução

O *Hardware* para computação de alto desempenho possui um alto valor agregado e normalmente necessita de uma configuração não trivial. É oneroso adquirir e gerenciar diferentes *hardwares* para cada componente em um sistema. Além do custo monetário, garantir a configuração correta para a sua aplicação é uma tarefa custosa que exige conhecimento técnico específico.

Uma alternativa para evitar tais custos é a virtualização, que consiste no processo de abstrair, por meio de *software*, componentes de *hardware* e sistema operacional. Dessa forma pode-se, com uma única máquina, criar diversas máquinas virtuais independentes, como se fossem, de fato, máquinas físicas diferentes. Esse processo ajuda a reduzir gastos monetários em compras de novos equipamentos e permite boa flexibilização, visto que, a qualquer momento, é possível acrescentar ou diminuir o número de máquinas, bem como alterar suas configurações.

Quando o objetivo é a replicação do ambiente de execução de um programa, sem necessariamente se abstrair o *hardware*, o uso de uma máquina virtual pode ser exagerado, acarretando o consumo de mais recursos que o necessário. Como alternativa, surgiu a conteinerização.

Os contêineres são considerados leves porque têm sobrecarga limitada em comparação com máquinas virtuais. Ao contrário da virtualização, os contêineres não exigem uma camada de emulação para serem executados. Em vez disso, eles usam o sistema operacional do *host*. Esse tipo de virtualização utiliza um único *kernel* para executar várias instâncias em um determinado sistema operacional. Desta maneira, é possível criar múltiplos sistemas isolados em um único *host* e acessar um único *kernel* [Dua et al. 2014].

Nesse cenário, há situações nas quais se deseja compartilhar o mesmo volume de arquivos entre diversas aplicações, executando em diferentes ambientes. Uma alternativa já existente há décadas é o uso do *Network File System*, ou *NFS*. Essa tecnologia permite que demais *hosts* conectados à rede possam acessar um mesmo diretório de arquivos. Outra alternativa é o uso da tecnologia de *Docker Volumes*, introduzida juntamente com a tecnologia de contêineres fornecida pelo *Docker*.

Devido às grandes dificuldades de se encontrar comparações entre as duas tecnologias, esse artigo verifica as diferenças de desempenho entre os dois métodos para compartilhamento de volumes. Saber essa diferença é fundamental para a computação de alto desempenho em projetos que buscam otimizar o compartilhamento de dados entre contêineres.

Após esta introdução, este artigo contém outras cinco seções. Na Seção 2 é feita uma breve explanação sobre os tipos de volumes para contêineres. Na Seção 3 encontra-se o planejamento do estudo experimental e na Seção 4 os resultados obtidos com esse estudo. Na seção 5 são descritos os trabalhos relacionados no contexto deste artigo e por fim, na Seção 6, são apresentadas as conclusões e trabalhos futuros.

2. Compartilhamento de Volume em Contêineres

Esta seção aborda as tecnologias para o compartilhamento de diretórios entre diferentes *hosts*.

2.1. NFS

O *NFS* é uma tecnologia para o compartilhamento de diretórios entre diferentes *hosts* (incluindo máquinas físicas diferentes) conectados à mesma rede. O *NFS* permite dois modos de operação: síncrono (o servidor só reconhecerá os dados depois que forem gravados em disco) e assíncrono (o servidor reconhecerá os dados antes de serem gravados em disco).

2.2. Docker Volumes

Docker Volumes é uma tecnologia para o compartilhamento de diretórios entre diferentes contêineres, executando dentro de uma mesma máquina física. Para o compartilhamento entre máquinas físicas diferentes é necessário o uso de outras ferramentas como, por exemplo, *Kubernetes Volumes*, que não pertence ao objetivo deste artigo e, portanto, não será abordado aqui.

O *Docker Volumes* também possui outros modos de operação: *consistent* (*host* e contêiner têm um visão idêntica da montagem o tempo todo), *cached* (permite atrasos antes que as atualizações no *host* apareçam no contêiner) e *delegated* (permite atrasos antes que as atualizações no contêiner apareçam no *host*), porém o objetivo do artigo é comparar apenas o modo *consistent*.

3. Planejamento do Estudo Experimental

Um planejamento de experimentos foi definido com o objetivo de comparar o desempenho entre os métodos descritos nas Seções 2.1 e 2.2. Os experimentos foram realizados em uma máquina

virtual (VM) com *Ubuntu Server* 18.04.3 LTS, contendo 512MB de memória, com o intuito de evitar caches e aumentar a quantidade de acessos ao disco. A VM foi executada em um notebook ASUS com Intel Core i7 - 3537U, 2.0GHz, 8GB de memória RAM e dois discos rígidos, sendo um com tecnologia HDD e outro com SSD.

Para realizar as medições, foi utilizado o *SysBench* como *benchmark* por ser o mais utilizado no levantamento apontado na Tabela 2. Operações de leitura e escrita, utilizando o *NFS* síncrono, assíncrono e volume do *Docker*, foram realizadas de acordo com a ordem abaixo:

- sem concorrência (um contêiner) em disco rígido paralelo (HDD)
- com concorrência (dez contêineres) em disco rígido paralelo (HDD)
- sem concorrência (um contêiner) em disco rígido serial (SSD)
- com concorrência (dez contêineres) em disco rígido serial (SSD)

Para o modo assíncrono, o *SysBench* permite a configuração da *flag async*, informando o sistema que utilizará cache e outros mecanismos de desempenho, conforme exemplo a seguir:

Trecho de código 1. Exemplo de comando do SysBench

```
sysbench --test=fileio --file-io-mode=async --file-total-size=2G
   --file-test-mode=rndrw --max-time=60 --max-requests=0 run
```

Para todos os experimentos foram utilizados arquivos com o tamanho de 2GB e o tempo de 60 segundos para cada execução. No total, foram realizadas 3100 medições para conseguir resultados precisos para geração dos gráficos e tabelas, apresentados na próxima Seção. Esses resultados englobam a quantidade de leitura, escrita e o *throughput*, realizadas no tempo de 60 segundos para cada execução.

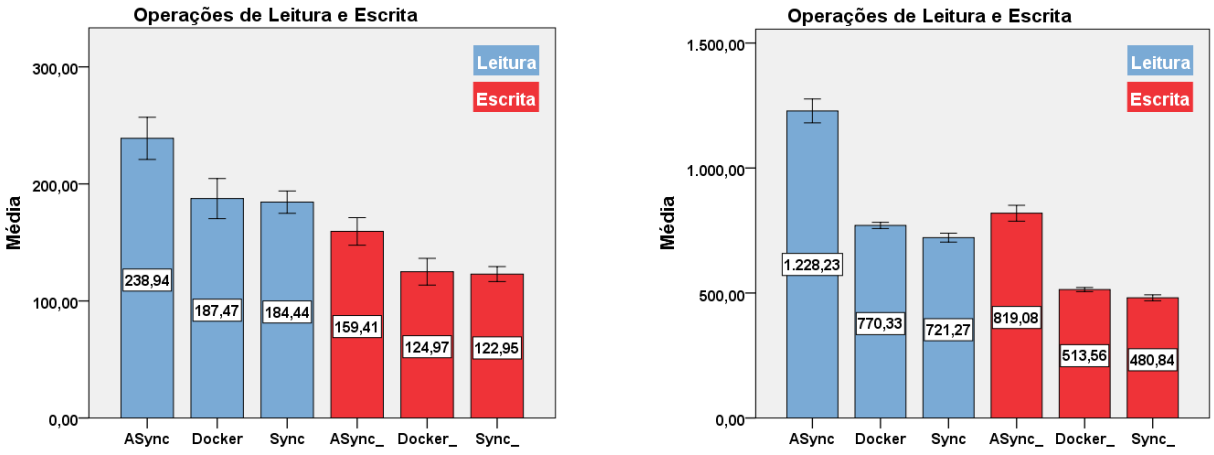
4. Resultados

A Tabela 1 mostra os dados estatísticos para todos os experimentos realizados. Conforme pode ser observado, todos os testes tiveram significância (p<0.05), com exceção das execuções entre o *Docker* e o *NFS* síncrono (leitura e escrita) para discos rígidos paralelos sem concorrência.

Tabela 1. Test-t das amostras

Tipo do experimento			**Média**	**Desvio Padrão**	**t**	**Sig**
HDD sem concorrência	Leitura	Docker - Sync	3,03	9,6252	1,724	0,095
		ASync - Docker	51,47567	11,78667	23,921	,000
	Escrita	Docker_ - Sync_	2,02067	6,41876	1,724	0,095
		ASync_ - Docker_	34,43533	7,74237	24,361	,000
SSD sem concorrência	Leitura	Docker - Sync	49,068	9,78441	27,468	,000
		ASync - Docker	457,8957	24,70485	101,518	,000
	Escrita	Docker_ - Sync_	32,712	6,52298	27,468	,000
		ASync_ - Docker_	305,5237	16,53264	101,219	,000
HDD com concorrência	Leitura	Sync - Docker	10,79887	5,16282	36,229	,000
		ASync - Docker	25,84617	10,36634	43,185	,000
	Escrita	Sync_ - Docker_	7,19927	3,44217	36,226	,000
		ASync_ - Docker_	17,27263	7,00878	42,685	,000
SSD com concorrência	Leitura	Sync - Docker	26,1093	10,13986	44,599	,000
		ASync - Docker	54,51083	10,93348	86,355	,000
	Escrita	Sync_ - Docker_	17,66503	6,79731	45,013	,000
		ASync_ - Docker_	36,51573	7,41558	85,289	,000

A Figura 1 apresenta a média das operações de leitura e escrita para o *Docker Volume* e *NFS*. Nesse caso, o *NFS* assíncrono tem melhor desempenho em comparação com o *NFS* síncrono e o volume do *Docker*.

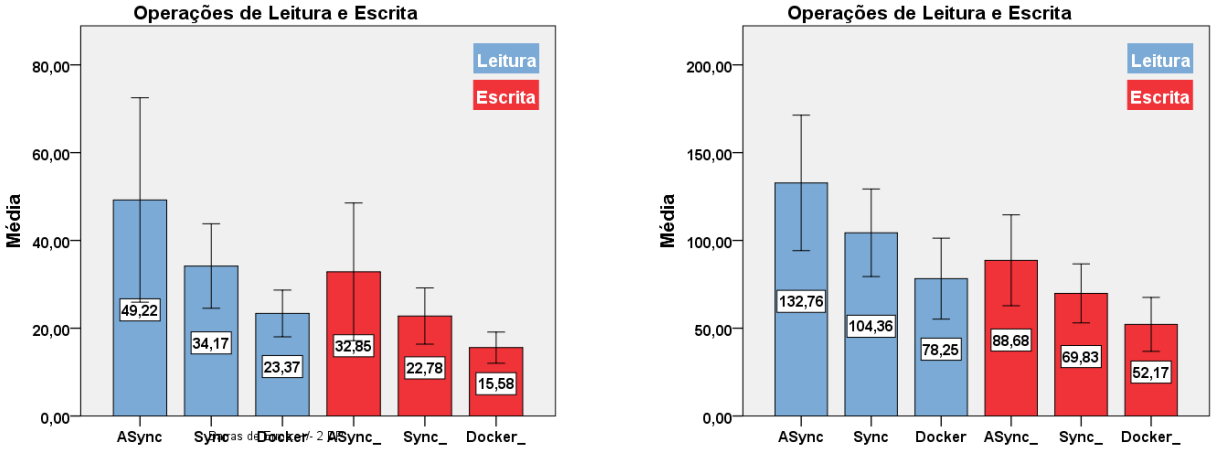


(a) Disco HDD (b) Disco SSD

Figura 1. Quantidade de operações de leitura e escrita sem concorrência

Ainda, o *Docker* mostrou melhor desempenho de leitura e escrita quando comparado com o *NFS* síncrono em discos rígidos seriais. Esse comportamento não foi observado para discos paralelos. Outro ponto a observar é um menor desvio padrão para discos rígidos seriais.

A Figura 2 mostra as execuções realizadas com concorrência entre os dez contêineres. Como visto, há um aumento do desvio padrão por esse motivo. Diferentemente da execução anterior, o *test-t* mostrou que há significância entre todos os testes realizados, tanto para discos paralelos quanto discos seriais.



(a) Disco HDD (b) Disco SSD

Figura 2. Quantidade de operações de leitura e escrita com concorrência

5. Trabalhos Relacionados

A Tabela 2 mostra os trabalhos que avaliam o desempenho de contêineres em relação à virtualização baseada em hipervisores e foi adaptada de [Bachiega et al. 2018].

Tabela 2. Trabalhos Relacionados

Artigo	Carga de Trabalho	Recursos					
		P	M	R	D	CV	O
[Che et al. 2010]	SPECCPU, LINPACK, RAMSPEED, LMbench, IOzone, Bonnie++, NetIO, WebBench, SysBench e SPECJBB	X	X	X	X		
[Xavier et al. 2013]	LINPACK, STREAM, IO-zone, NetPIPE, NPB e IBS	X	X	X	X		
[Babu et al. 2014]	UnixBench						X
[Felter et al. 2015]	PXZ, LINPACK, STREAM, nuttcp, netperf, FIO, Redis e SysBench	X	X	X	X		X
[Ruiz et al. 2015]	NPB e TAU			X			X
[Adufu et al. 2015]	autodock3		X				
[Amaral et al. 2015]	CPU-intensive, Sysbench e netperf	X	X	X			X
[Beserra et al. 2015]	HPL e NetPIPE	X		X			X
[Joy 2015]	JMeter						X
[Morabito et al. 2015]	Y-cruncher, NBENCH, Geekbench, noploop, Linpack, Bonnie++, Sysbench, IOzone, STREAM e netperf	X	X	X	X		
[Xavier et al. 2015]	Swingbench e Sysbench	X	X		X		
[Chung et al. 2016]	HPL e Graph500		X				X
[Jaikar et al. 2016]	HTCondor	X	X	X			
[Ruan et al. 2016]	SPEC CPU 2006, STREAM, FIO, netperf e HiBench	X	X	X	X		
[Herbein et al. 2016]	LINPACK	X		X	X		
[Barik et al. 2016]	AIO Stress, Ram-speed, IOzone, Tbench, iperf, RuBBoS, ApacheBench, Blake2, 7-zip e OpenSSL benchmark		X	X	X		X
[Beserra et al. 2015]	fs test				X		X
[Eiras et al. 2016]	ApacheBench e Stress Tool			X			X
[Kozhirbayev and Sinnott 2017]	Y-cruncher, LINPACK, Geekbench, Bonnie++, Sysbench, STREAM, netperf e iperf	X	X	X	X		
[Li et al. 2017]	Iperf, HardInfo, Bonnie++ e STREAM	X	X	X	X		
[Mavridis and Karatza 2017]	LINPACK, STREAM, IO-zone e netperf	X	X	X	X		
[Lingayat et al. 2018]	Ferramentas do Docker e Linux	X		X	X		
[Zeng et al. 2017]	Ferramentas do Docker e Linux			X			
[Mizusawa et al. 2018]	Ferramentas do Docker e Linux				X		

A tabela foi organizada pelo tipo de carga de trabalho utilizada e quais os recursos que foram testados: (P) Processamento, (M) Memória, (R) Rede, (D) Disco, (CV) Compartilhamento de Volume, (O) Outros, como cópia de arquivo, criação de processos, número de instâncias, entre outros.

Os trabalhos [Che et al. 2010], [Xavier et al. 2013], [Babu et al. 2014], [Felter et al. 2015], [Ruiz et al. 2015], [Amaral et al. 2015], [Beserra et al. 2015], [Joy 2015], [Morabito

et al. 2015], [Xavier et al. 2013] e [Li et al. 2017], [Lingayat et al. 2018] mostram análises de desempenho de virtualização baseada em contêineres e em hipervisores. Para a maioria dos casos, os contêineres têm melhor avaliação para processamento, memória, rede e disco.

Os artigos de [Adufu et al. 2015] e [Jaikar et al. 2016] abordam a utilização de contêineres para trabalhos científicos, em que estes também têm melhor desempenho. [Chung et al. 2016] e [Beserra et al. 2016] abordam trabalhos que avaliam o desempenho de contêineres para Computação de Alto Desempenho, tornando uma alternativa viável para estas aplicações.

Os trabalhos de [Ruan et al. 2016], [Barik et al. 2016], [Kozhirbayev and Sinnott 2017] tratam de avaliar desempenho dos contêineres para ambientes de nuvem, assim como [Herbein et al. 2016], que trata de avaliação de desempenho de nuvem de contêineres para aplicações de alto desempenho. [Eiras et al. 2016] aborda a avaliação de desempenho de um *proxy* entre KVM e Docker. [Mavridis and Karatza 2017] trata da avaliação de desempenho de contêineres sendo executados dentro de uma máquina virtual.

O artigo [Mizusawa et al. 2018] realiza a avaliação de desempenho de operações de arquivo em contêineres. Por fim, o trabalho de [Zeng et al. 2017] investiga e analisa o status atual do desenvolvimento da rede de contêineres.

Dos trabalhos aqui relacionados, a maioria apresenta uma vantagem de desempenho dos contêineres em relação à virtualização baseada em hipervisores. Entretanto, o trabalho de [Barik et al. 2016] concluiu que contêineres têm pior desempenho do que um hipervisor quando há alta taxa de pacotes de dados na rede, assim como o artigo de [Ruan et al. 2016] que concluiu que a execução de contêineres em máquinas virtuais resulta em uma degradação de desempenho de E/S de disco. Ainda, é possível observar que o *SysBench* foi o *benchmark* mais utilizado e nenhum dos trabalhos encontrados avaliou o desempenho de contêineres quanto ao compartilhamento de volume, foco deste trabalho.

6. Conclusões

Este artigo teve por objetivo mostrar a avaliação de desempenho de compartilhamento de volume utilizando o *Docker Volume* e *NFS*. Conforme pode ser observado na Seção 5, muitos trabalhos realizaram a avaliação de desempenho de contêineres sob diferentes aspectos, entretanto, até o momento, nenhum trabalho prévio havia feito a avaliação com o compartilhamento de volume.

Conforme pode ser obervado nos resultados, e usando como base a preparação dos experimentos, chegou-se à conclusão que, em termos de dificuldade para configuração, utilizar *Docker Volumes* é um pouco mais simples quando comparado à utilização do *NFS*. Em termos de desempenho, o *NFS* assíncrono mostrou melhor desempenho em todos os testes realizados, tanto para operações de escrita e leitura em discos paralelos e seriais. Porém, cabe ressaltar que desabilitar o mecanismo de sincronização possui um impacto muito positivo sobre o desempenho, porém aumenta os riscos de perda de dados em casos de *hard resets* ou *crashes* no *host*.

Ademais, conforme apresentado na Tabela 1, as amostras com o *test-t* demonstraram que todos as execuções tiveram significância, com exceção dos testes de leitura e escrita para discos paralelos sem concorrência. Dessa forma, esses resultados podem contribuir para a elaboração de projetos que visam desempenho ou a escolha de modos de operação de compartilhamento de volumes.

Para trabalhos futuros, outros testes poderão ser realizados com tamanhos de arquivos di-

ferentes ou outros modo de operação do Docker Volume, como por exemplo, *cached* e *delegated* que utilizam mecanismos parecidos com o *NFS* assíncrono.

Referências

Adufu, T., Choi, J., and Kim, Y. (2015). Is container-based technology a winner for high performance scientific applications? In *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 507–510.

Amaral, M., Polo, J., Carrera, D., Mohomed, I., Unuvar, M., and Steinder, M. (2015). Performance evaluation of microservices architectures using containers. In *2015 IEEE 14th International Symposium on Network Computing and Applications*, pages 27–34.

Babu, A., J., H. M., Martin, J. P., Cherian, S., and Sastri, Y. (2014). System performance evaluation of para virtualization, container virtualization, and full virtualization using xen, openvz, and xenserver. In *2014 Fourth International Conference on Advances in Computing and Communications*, pages 247–250.

Bachiega, N. G., Souza, P. S. L., Bruschi, S. M., and d. R. S. de Souza, S. (2018). Container-based performance evaluation: A survey and challenges. In *2018 IEEE International Conference on Cloud Engineering (IC2E)*, pages 398–403.

Barik, R. K., Lenka, R. K., Rao, K. R., and Ghose, D. (2016). Performance analysis of virtual machines and containers in cloud computing. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 1204–1210.

Beserra, D., Moreno, E. D., Endo, P. T., and Barreto, J. (2016). Performance evaluation of a lightweight virtualization solution for hpc i/o scenarios. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 004681–004686.

Beserra, D., Moreno, E. D., Endo, P. T., Barreto, J., Sadok, D., and Fernandes, S. (2015). Performance analysis of lxc for hpc environments. In *2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems*, pages 358–363.

Che, J., Shi, C., Yu, Y., and Lin, W. (2010). A synthetical performance evaluation of openvz, xen and kvm. In *2010 IEEE Asia-Pacific Services Computing Conference*, pages 587–594.

Chung, M. T., Quang-Hung, N., Nguyen, M. T., and Thoai, N. (2016). Using docker in high performance computing applications. In *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, pages 52–57.

Dua, R., Raja, A. R., and Kakadia, D. (2014). Virtualization vs containerization to support paas. In *2014 IEEE International Conference on Cloud Engineering*, pages 610–614.

Eiras, R. S. V., Couto, R. S., and Rubinstein, M. G. (2016). Performance evaluation of a virtualized http proxy in kvm and docker. In *2016 7th International Conference on the Network of the Future (NOF)*, pages 1–5.

Felter, W., Ferreira, A., Rajamony, R., and Rubio, J. (2015). An updated performance comparison of virtual machines and linux containers. In *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 171–172.

Herbein, S., Dusia, A., Landwehr, A., McDaniel, S., Monsalve, J., Yang, Y., Seelam, S., and Taufer, M. (2016). Resource management for running hpc applications in container clouds. *Lecture Notes in Computer Science*, 9697:261–278.

Jaikar, A., Shah, S., Bae, S., and Noh, S. (2016). Performance evaluation of scientific workflow on openstack and openvz. *Social-Informatics and Telecommunications Engineering, LNICST*, 167:126–135.

Joy, A. M. (2015). Performance comparison between linux containers and virtual machines. In *2015 International Conference on Advances in Computer Engineering and Applications*, pages 342–346.

Kozhirbayev, Z. and Sinnott, R. (2017). A performance comparison of container-based technologies for the cloud. *Future Generation Computer Systems*, 68:175–182.

Li, Z., Kihl, M., Lu, Q., and Andersson, J. A. (2017). Performance overhead comparison between hypervisor and container based virtualization. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 955–962.

Lingayat, A., Badre, R. R., and Kumar Gupta, A. (2018). Performance evaluation for deploying docker containers on baremetal and virtual machine. In *2018 3rd International Conference on Communication and Electronics Systems (ICCES)*, pages 1019–1023.

Mavridis, I. and Karatza, H. (2017). Performance and overhead study of containers running on top of virtual machines. In *2017 IEEE 19th Conference on Business Informatics (CBI)*, volume 02, pages 32–38.

Mizusawa, N., Kon, J., Seki, Y., Tao, J., and Yamaguchi, S. (2018). Performance improvement of file operations on overlayfs for containers. In *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 297–302.

Morabito, R., Kjällman, J., and Komu, M. (2015). Hypervisors vs. lightweight virtualization: A performance comparison. In *2015 IEEE International Conference on Cloud Engineering*, pages 386–393.

Ruan, B., Huang, H., Wu, D., and Jin, H. (2016). A performance study of containers in cloud environment. *In: Wang G., Han Y., Martínez Pérez G. (eds) Advances in Services Computing. APSCC 2016.*, 10065:343–356.

Ruiz, C., Jeanvoine, E., and Nussbaum, L. (2015). Performance evaluation of containers for hpc. In *In: Hunold S. et al. (eds) Euro-Par 2015: Parallel Processing Workshops. Euro-Par 2015*, pages 813–824.

Xavier, M. G., Neves, M. V., Rossi, F. D., Ferreto, T. C., Lange, T., and Rose, C. A. F. D. (2013). Performance evaluation of container-based virtualization for high performance computing environments. In *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 233–240.

Xavier, M. G., Oliveira, I. C. D., Rossi, F. D., Passos, R. D. D., Matteussi, K. J., and Rose, C. A. F. D. (2015). A performance isolation analysis of disk-intensive workloads on container-based clouds. In *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 253–260.

Zeng, H., Wang, B., Deng, W., and Zhang, W. (2017). Measurement and evaluation for docker container networking. In *2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 105–108.