Literature Review: Performance Comparison of Virtual Machines and Linux Containers

# Felter, W., Ferreira, A., Rajamony, R., & Rubio, J. (2015)

## “An Updated Performance Comparison of Virtual Machines and Linux Containers”

## Introduction

Virtualization has long been a core enabler of modern computing environments, particularly within enterprise data centers and cloud computing platforms. Historically, Virtual Machines (VMs) have provided hardware-level abstraction, allowing multiple operating systems to run concurrently on the same physical hardware while maintaining strong isolation. This isolation is achieved through a hypervisor, which manages the hardware resources and ensures that each VM operates independently.

However, the rise of Linux Containers (LCs) has introduced a paradigm shift by offering a more lightweight form of virtualization, one that operates at the operating system level rather than emulating hardware. Containers share the host OS kernel and isolate applications using namespaces and control groups (cgroups), which results in lower overhead compared to VMs.

This shift has become increasingly relevant with the growing adoption of cloud-native architectures, DevOps practices, and microservices. Containers, exemplified by technologies like Docker and Kubernetes, promise reduced overhead, faster startup times, and seamless portability. These benefits are particularly advantageous in environments that require rapid scaling, continuous integration/continuous deployment (CI/CD), and efficient resource utilization. Yet, a critical question remains: do these benefits come at the cost of performance?

The paper by Felter et al. (2015) is a timely and pivotal contribution to this ongoing debate. It systematically compares the performance of VMs and containers across multiple dimensions using rigorous benchmarking techniques. The motivation lies in the need for evidence-based decision-making: developers, system administrators, and architects require concrete data to decide when to use VMs or containers, particularly in environments where performance, scalability, and efficiency are key concerns.

This study also addresses a gap in existing research. While earlier works focused either on isolated aspects of performance or theoretical frameworks, Felter et al. aim to provide a comprehensive, empirical performance comparison, thereby supporting practical infrastructure planning and deployment strategies.

## Contributions

Felter et al. (2015) make several noteworthy contributions that enhance the current understanding of virtualization technology in high-performance and production environments:

* **Comprehensive Benchmark Suite**: The authors employ a diverse set of industry-standard benchmarks, such as SPEC CPU2006, IOzone, Netperf, and STREAM, covering CPU, memory, disk, and network performance. This multi-dimensional approach ensures that the findings are not biased toward any single type of workload. By using a variety of benchmarks, the study provides a holistic view of how VMs and containers perform under different conditions.
* **Equal Hardware and Configuration**: A significant strength of the study lies in its use of a controlled experimental setup. Both the KVM-based VM and Docker-based container were allocated equal shares of CPU, memory, and storage resources. This ensures fairness and reproducibility in performance comparisons. The controlled environment eliminates variables that could skew the results, providing a clear comparison between the two virtualization technologies.
* **Granular Analysis of Virtualization Overheads**: The paper goes beyond just comparing raw performance; it investigates the reasons behind performance differences. For example, it explains how the additional layers of abstraction in VMs (e.g., hypervisor, virtualized I/O stack) can introduce measurable delays in I/O operations. This detailed analysis helps in understanding the trade-offs involved in choosing between VMs and containers.
* **Demonstration of Container Efficiency**: The research concludes that containers either match or exceed VM performance in most cases. Particularly in memory throughput and disk I/O, containers outperform VMs due to their more direct access to host OS resources. This efficiency makes containers a compelling choice for applications that require high performance and low latency.
* **Practical Utility**: The findings directly aid decision-making in modern IT environments, especially for organizations seeking cost savings, energy efficiency, and high-speed deployment. The empirical data provided by the study can be used to optimize infrastructure and improve the performance of applications.

Overall, the paper not only contributes empirical data but also offers a strong analytical framework that can be used for further research or enterprise evaluation.

## Results

The experimental results are a central strength of the study, revealing how containers and VMs perform under real-world conditions. Below is a breakdown of the performance results across different dimensions:

* **CPU Performance**: Using the SPEC CPU2006 benchmark, the researchers observed only a marginal performance difference between VMs and containers. Containers demonstrated a slightly better throughput, attributed to the absence of hypervisor-related overheads. However, the difference was not drastic, indicating that modern VMs are still highly optimized for CPU-bound tasks. This suggests that for CPU-intensive applications, both VMs and containers can be viable options.
* **Memory Performance**: For memory bandwidth and latency testing, the STREAM benchmark was used. Containers consistently outperformed VMs due to fewer abstraction layers and closer integration with the host kernel's memory management systems. This makes containers more suitable for memory-intensive applications, such as big data analytics or in-memory databases. The reduced latency and higher throughput of containers can significantly enhance the performance of such applications.
* **Disk I/O Performance**: Using IOzone, the authors tested both random and sequential read/write operations. Here, containers exhibited up to 30% faster performance than VMs in random I/O scenarios. The VM's virtualized disk drivers and emulated I/O paths introduced bottlenecks, especially under high concurrency. This performance gap makes containers more attractive for applications requiring fast access to storage, such as log processors or file-based caching systems. The improved I/O performance of containers can lead to faster data processing and reduced latency.
* **Network Performance**: Network throughput and latency were tested using Netperf. In low-latency operations (e.g., TCP\_RR), both VMs and containers performed similarly. However, under high throughput conditions, containers had a slight advantage. The more streamlined network stack of Docker, coupled with lower packet processing overhead, resulted in better scalability. This makes containers a better choice for network-intensive applications, such as web servers and microservices.
* **Boot Time and Efficiency**: The study also included boot time analysis. Containers launched in sub-second durations, while VMs took several seconds to minutes, depending on the OS. This makes containers especially appealing for on-demand scaling and CI/CD environments, where startup latency is a critical factor. The rapid startup times of containers enable faster deployment and scaling, improving the overall efficiency of development and operations workflows.

## Discussion

The findings of the study by Felter et al. have far-reaching implications for cloud infrastructure design, DevOps workflows, and performance optimization.

* **Impact and Relevance**: This paper arrives at a time when enterprises are making strategic decisions about containerization. Its conclusion — that containers are not only more efficient but also match or outperform VMs — validates the movement toward container-first deployments in cloud-native applications. Technologies like Docker, Kubernetes, and container orchestrators benefit directly from such performance findings. The study provides concrete evidence supporting the adoption of containers, helping organizations make informed decisions.
* **Strengths**:
  + **Methodological Soundness**: The study uses a fair and reproducible methodology. Unlike many earlier papers that relied on anecdotal evidence or simulations, this research is grounded in real benchmark data. The rigorous approach ensures the reliability and validity of the findings.
  + **Clarity and Accessibility**: The paper is written clearly, making complex virtualization concepts accessible to practitioners and academics alike. The straightforward presentation of results and analysis helps readers understand the implications of the study.
  + **Practical Contribution**: It answers a real-world question with data that can be directly applied in infrastructure planning. The insights provided by the study can guide the optimization of IT environments and improve the performance of applications.
* **Limitations**:
  + **Security and Isolation Not Covered**: One of the main criticisms is the lack of analysis on security and tenant isolation. While containers perform well, they are often perceived to be less secure than VMs due to shared kernel usage. Future research should address these concerns to provide a more comprehensive evaluation.
  + **Hardware Scope**: All tests were conducted on a single hardware configuration. Future research should validate these results across heterogeneous systems and varying workloads. This would ensure the generalizability of the findings and provide a broader understanding of performance differences.
  + **No Multi-Tenancy Stress Test**: The study does not examine how containers behave under multi-tenant conditions, which is crucial for cloud service providers. Evaluating the performance and isolation of containers in multi-tenant environments would provide valuable insights for service providers.
* **Future Work**:
  + **Security Benchmarking**: Future studies could evaluate how containers compare to VMs in terms of isolation, privilege escalation, and kernel attack surface. This would address the security concerns associated with containerization and provide a more comprehensive comparison.
  + **Hybrid Virtualization Models**: Research can also investigate hybrid environments, such as containers within VMs (as in many Kubernetes-on-VM setups), and how that impacts performance and security. Understanding the trade-offs and benefits of hybrid models would help organizations optimize their infrastructure.
  + **Edge and IoT Scenarios**: Exploring how containers and VMs perform in constrained environments like edge computing or embedded devices can further extend the practical value of this work. Evaluating the performance and efficiency of virtualization technologies in these scenarios would provide insights for deploying applications in resource-constrained environments.