Llenard C. Diama
IV-BCSAD

A Narrative Report about Virtualization vs Containerization

Introduction

This report summarizes all the findings and methodologies in regards with the study of the school of computing of the University of the West of Scotland. The research addresses the critical decision between virutalization and containerization technologies for deploying applications, especially withing the resource constrained environment of Edge Computing.

Overview and Motivation

In the scope of the Computing Continuum, from Cloud to Edge/Fog, the evolving adoption of such advanced technologies as e-Government, Industry 4.0, and Smart Cities, which are powered by AI, requires a critical review of the underlying infrastructure in terms of software deployment. The decision between virtualization and containerization will be crucial for application performance, scalability, and efficiency.

The Study's Methodology

The study evaluated two deployment methods using criteria such as compatibility, scalability, and energy efficiency, with a focus on power-efficient Edge Computing. Testing was conducted on an ARM-based Raspberry Pi 4B+ running 64-bit Raspbian 12 and an x86 environment hosted on Proxmox VE. Tools included Stress-ng for applying controlled CPU, memory, and I/O workloads measured through "Bogo Ops," Top System Monitor for real-time performance metrics captured as JSON, and a UM24C power meter for logging voltage, current, and overall power consumption to assess system sustainability.

The Test Architecture

The comparison included various configurations of container runtimes and managers, along with different virtual machine types171717:

Containerization Stacks:

- Kubernetes (minikube) with Podman container runtime18.
- Docker Engine with Docker Compose orchestration19.
- Other couplings using Kubernetes, Podman, and Docker Engine20.

Virtualization Stacks:

- x86 VM (64-bit QEMU with qemu-system-x86_64) to test emulation on an ARM Environment.
- ARM VM(64-bit QEMU with qemu-system-aarch64 and KVM extension)

Key Findings and Result

Performance Comparison
Testing showed that for short-term, single-node use with only a few containers, the Docker Compose and Podman coupling generally performed better across CPU, I/O, and memory metrics.

- CPU Performance: For short-term tests, such as a 5% difference in performance for CPU usage, Kubernetes solutions were less effective because of higher overheads and complexity. Their performance improved in the longer test period of 10 minutes and thus showed better long-term performance.

- I/O Disk Performance: Docker Compose was, on average, 40% more efficient than Kubernetes for the 1-minute test. That could be because of Podman's daemon-less operation, preventing network service overload, whereas that of Kubernetes operates several complex components: Kuberlet, API server, etc.

- Memory Performance: The differences were relatively smaller compared to the CPU and I/O, though still in favor of the lightweight Docker Compose and Podman stack.

Power Efficiency and Architecture

- The general test comparison among those executed on the testbed Raspberry Pi evidenced the inefficiency of ×86 emulation.

- The rpi, Docker, and aarch64 VM measurements were very close, highlighting the high efficiency of containers as well as the suitability of virtualization with appropriate extensions (KVM).

- The study concluded that in terms of energy consumption, containerization is the most ecologically advantageous option.

- When executed over a longer duration, 20 minutes, the power efficiency (Bogo Ops/watt second) of Kubernetes was greater than that of Docker.

Use Case Applicationm

The best architectural choice has been utilized to illustrate one use case: dynamic point-cloud generation using Open Drone Map from drone imagery.

- It uses a Raspberry Pi for preprocessing and an Edge server with GPU acceleration to compute a transformation matrix.

- Kubernetes orchestrates Docker containers in the Cloud, dynamically manages and scales ODM worker instances to process different spectral bands in parallel.

- This distributed processing has helped to speed up the entire process. The processing times of pointcloud generation varied from 858.71 s to 912.28 s across the four different spectral bands.

Conclusion

The choice between virtualization and containerization is not straightforward and depends on specific requirements.

- **Containers** are recommended when flexibility and resource efficiency are critical, particularly for resource-constrained Edge Computing environments. They are ideal for separating apps on the same host without the overhead of a virtual OS. For one-time tasks requiring resource efficiency (like training AI models), Docker Compose and Podman are recommended.

- **Virtual Machines** are relevant when complete isolation, specific OS configurations, hardware device granularity (e.g., USB sticks), or a graphical user interface for monitoring are required.