Performance Analysis of Live Migration using qemu kvm

CS695 Project Report

Pranab Kumar Paul (23M0800)
Debdoot Roy Chowdhury (23M0765)

May 2024

Contents

1 Background Study

Live migration of virtual machines (VMs) refers to the process of transferring a running VM from one physical host to another without causing noticeable downtime or service interruption. The term "live" in live migration signifies that the VM remains operational and accessible throughout the migration process, ensuring continuous availability of services to end-users. There are two primary methods of live migration: *pre-copy* and *post-copy*. During the Iterative Pre-copy phase of live migration, the virtual machine's memory is transferred from source to destination in multiple iterations. Initially, all memory pages are copied, followed by subsequent iterations transferring only modified pages (dirty pages).In the Post-copy method of live migration, the virtual machine's execution is quickly transferred to the destination host with minimal initial memory state. Subsequently, missing memory pages are transferred in the background while the VM continues running. Live migration enhances fault tolerance by enabling rapid VM relocation during hardware failures, minimizing service disruptions. It optimizes resource utilization by balancing workloads across hosts, improving system efficiency. Additionally, live migration supports energy efficiency through VM consolidation and dynamic power management, reducing data center power consumption.In modern computing, live migration is widely used in cloud environments for workload balancing, maintenance, and resource optimization. It supports seamless upgrades and maintenance of virtualized infrastructure, ensuring continuous service availability. Live migration is also integral to disaster recovery strategies, enabling quick VM relocation to backup sites during emergencies.

2 Study of Effectiveness of Live Migration

The experiment aims to evaluate the effectiveness of live migration in a virtualized environment. It involves migrating running virtual machines (VMs) between physical hosts while monitoring various performance metrics. The experiment will measure downtime during migration, assessing the impact on service availability. Additionally, resource utilization before, during, and after migration will be analyzed to understand efficiency gains. By conducting controlled migrations under different workload scenarios, the experiment aims to quantify the benefits of live migration in terms of fault tolerance, resource optimization, and energy efficiency. The experiment will utilize tools like qemu kvm to orchestrate migrations and collect performance data. Results will provide insights into the practical advantages and limitations of live migration, informing best practices for optimizing virtualized environments.

3 Experiment Setup

The experimental setup involved utilizing two distinct computing platforms: a laptop equipped with an AMD Ryzen 5600H CPU, 8GB of RAM, and a 512GB NVMe SSD; and a laboratory desktop featuring an Intel Core i5 13th generation processor, 16GB of RAM, and a 1TB SSD. The network connectivity for the desktop was established with a maximum bandwidth of 1Gbps. To facilitate live migration experiments, the QEMU + KVM virtualization setup was employed. Furthermore, a network-shared storage solution was utilized to enhance the efficiency and effectiveness of live migration procedures within the experimental framework.

Another experiment setup was done with two lab computers with configuration: CPU : Intel i5 13th generation, Memory: 16GB RAM and 1 TB SSD and both with operating system ubuntu 22.04 LTS. All the experiments with this setup is mentioned specifically in the figures from figure 3.

4 Implementation Details

The experiment incorporated Network Address Translation (NAT) and leveraged Network File System (NFS) for optimizing live migration efficiency. Initially, experiments were conducted without load testing to evaluate the efficacy of pre-copy and stop-and-copy methods. Subsequently, load testing was implemented using a custom bash script that initialized and iteratively updated a large array to simulate varying levels of memory page modifications. This approach allowed control over array size and inter-update intervals (i.e., dirty rate) to influence the rate of memory page changes during migration. Various experimental parameters including VM size, network bandwidth, and dirty page rate were systematically adjusted to measure performance metrics such as downtime duration, total migration time, and pre-copy phase duration. Detailed analysis of these metrics is presented in the Experiment and Evaluation section.

5 Experiment and Evaluation

Below are the results of several experiments conducted with various parameters and performance metrics.

5.1 Network Bandwith vs Pre-Copy time Analysis

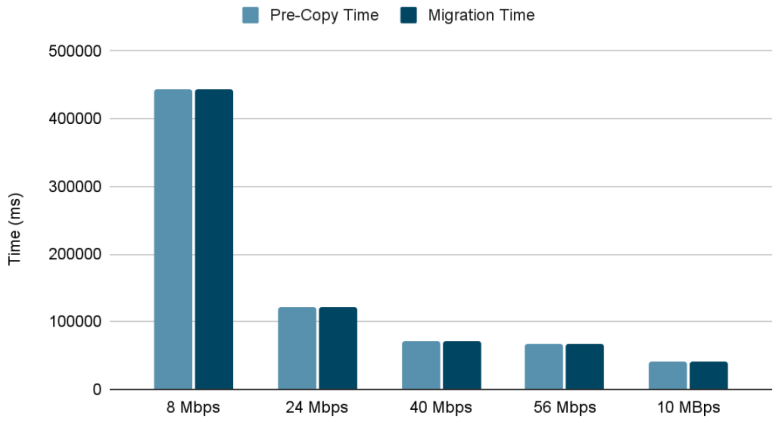

Figure 1: Performance Analysis of pre-copy time with varying VM Size

Observation: By observing the effect of varying network bandwidth, it becomes evident that the duration of the pre-copy phase and the overall migration time exhibit similar trends, with the migration time slightly exceeding the pre-copy time due to associated downtime (Total Migration Time = Pre-Copy Time + Overhead). The graphical representation suggests that increasing network bandwidth leads to a notable decrease in downtime and total migration time. This indicates that higher network bandwidth contributes significantly to reducing migration-related delays and enhancing overall efficiency.

5.2 Pure Stop-and-Copy Migration Analysis

Observation: When varying the VM size, the observation in figure 2 reveals a direct relationship between VM size, stop-and-copy migration time, and downtime. As VM size increases, both stop-and-copy migration time and downtime also increase. This trend is particularly notable because in stop-and-copy migration, the VM is entirely halted before being migrated to the destination, resulting in extended migration durations that closely mirror downtime periods. The findings underscore the impact of

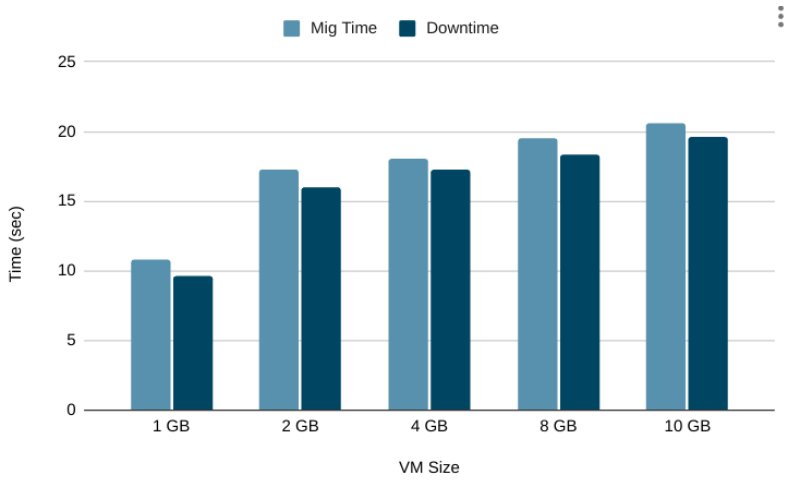

Figure 2: Performance Analysis of stop-and-Copy Downtime and Migration time with varying VM Size

VM size on migration efficiency, highlighting the trade-off between larger VMs and increased migration-related delays in stop-and-copy scenarios.

5.3 Pre-Copy Migration Analysis

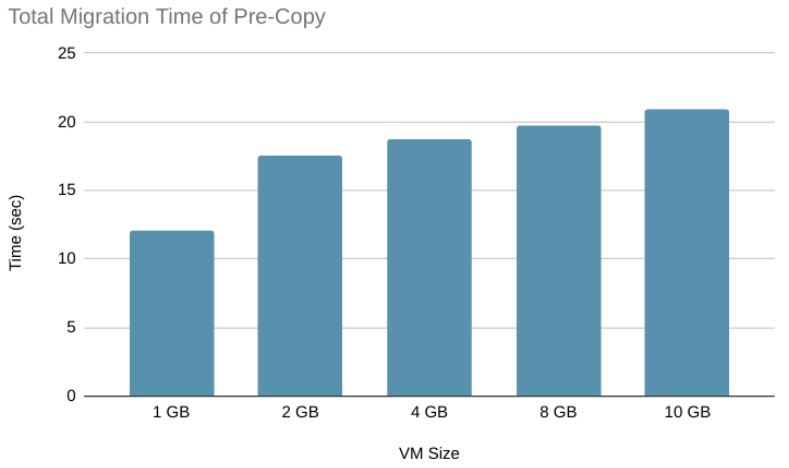

Figure 3: Performance Analysis of Pre-Copy Migration time with varying VM Size

Observation: Observations in figure 3 and 4 reveal that as VM size increases, both pre-copy migration time and downtime escalate. Comparing stop-and-copy with pre-copy migration, stop-and-copy demonstrates notably lower downtime due to the iterative nature of pre-copy, involving multiple memory copies. This highlights a trade-off between migration efficiency and downtime in virtualized environments. Optimizing pre-copy parameters is crucial for mitigating downtime and enhancing migration performance, underscoring the complexities of VM migration strategies.

5.4 Performance Analysis between Pre-Copy and Stop-and-Copy

Observation: As depicted in Figure 5, the total migration time for stop-and-copy is consistently shorter across different VM sizes. This efficiency is attributed to the

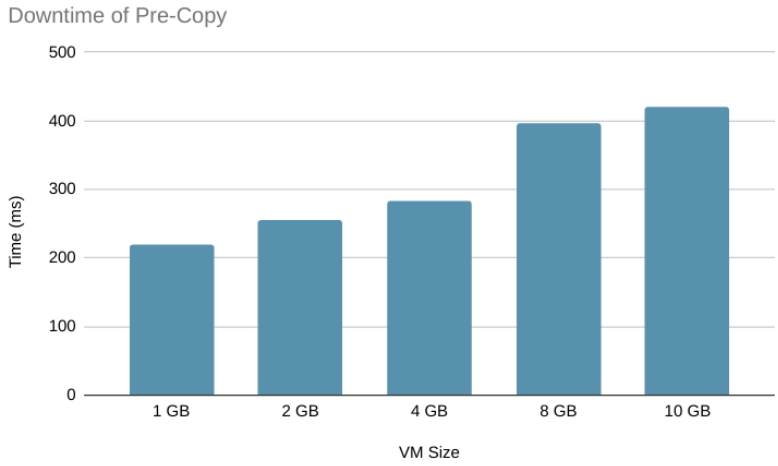

Figure 4: Performance Analysis of Pre-Copy Downtime time with varying VM Size

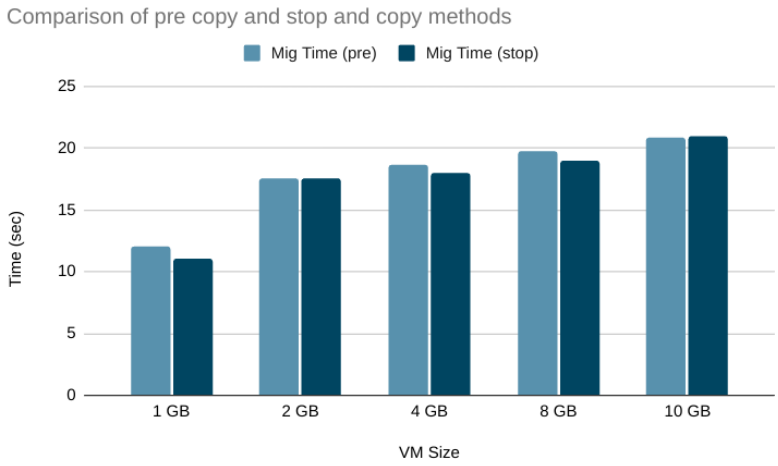

Figure 5: Performance Analysis of Pre-Copy and Stop-and-Copy Migration time with varying VM Size

iterative nature of pre-copy migration, where the iterative pre-copy phase involves repeated transfers of dirty pages to the destination VM, resulting in prolonged migration durations. The comparison underscores the inherent efficiency of stop-and-copy migration, highlighting the iterative challenges associated with pre-copy migration strategies.

5.5 Proof of loadtest

We want to prove that our loadtest is working fine and is suitable for the experiments below. For assuring that we first change the working set or the size allocated to the loadtest program and similarly observe the Dirty rate using the virsh command. The observations are given in figure 7.

Observation: From the above figure we can observe that the Dirty Page rate is increasing with increase with the memory allocation to our loadtest program. Hence the loadtest is working fine and it is certain that the dirty rate of the program (Number of pages overwritten per unit time) which is measured in pages/sec in the virsh is suitable for our experiments.

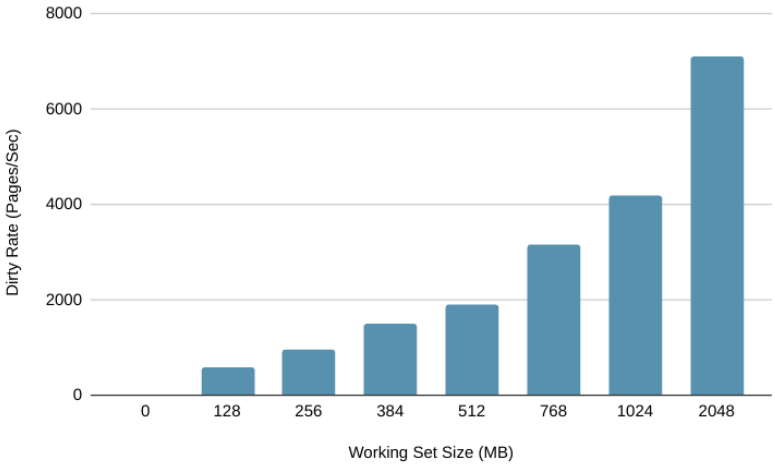

Figure 6: Proof of correctness using working set and dirty page rate

5.6 Analysis of Pre-Copy Method with Dirty Page Rate

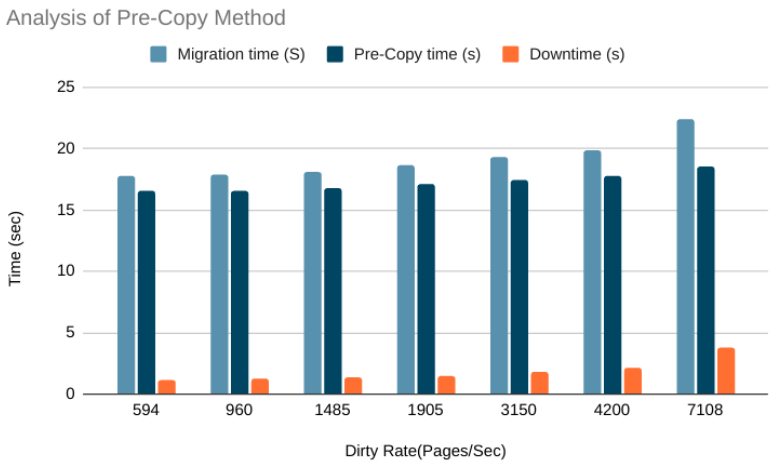

Figure 7: Analysis of Pre-Copy Method with Dirty Page Rate

Observation: From the above plot we can observe with increase the Dirty Page rate The total migration time, pre-copy time and also the downtime is also increasing. The total migration time is the sum of pre-copy time and downtime. It can be observed from the graph that the gap between the Total migration time and pre-copy time is equal to the downtime.

6 Conclusion

In conclusion, the performance evaluation of live migration using QEMU and KVM highlights its significant advantages and challenges in virtualized environments. Our experiments demonstrated the effectiveness of live migration for workload management, resource optimization, and fault tolerance. We observed that both pre-copy and stop-and-copy methods offer distinct benefits depending on workload characteristics and migration requirements.

Optimizing VM size, network bandwidth, and dirty page rate is crucial for minimizing downtime and ensuring smooth migration. Addressing challenges such as performance overhead and network bottlenecks through fine-tuning is essential for maximizing live migration benefits. This project underscores live migration's significance in virtual environments, providing flexibility, scalability, and improved resource utilization, contributing to ongoing virtualization research and advancements.