A Performance Comparison of In-Memory Virtual Desktop Environment

Dai Hoang Tran
Department of
Computer Engineering
Kyung Hee University
Email: dai.tran@khu.ac.kr

Tien-Dung Nguyen
Department of
Computer Engineering
Kyung Hee University
Email: ntiendung@khu.ac.kr

Eui-Nam Huh
Department of
Computer Engineering
Kyung Hee University
Email: johnhuh@khu.ac.kr

Choong Seon Hong
Department of
Computer Engineering
Kyung Hee University
Email: cshong@khu.ac.kr

***Abstract*—In-memory Computing (IMC) is the new trend for enabling high-performance computation and fast data processing. It is currently being used for large enterprises, e-commerce shops who need real-time interactions, low latency responses and instant results. Given the enhancement of the IMC, we apply this new paradigm to the Virtual Desktop Environment (VDE), and look into the performance differences in comparison with traditional VDE. The end results shows positive feedback, but there are trade-offs we need to concern for the In-memory Virtual Desktop Environment.**

I. Introduction

With the advance of technology, modern business information system nowadays consumes and processes huge amount of data in their daily operations. Not only that, but the end-users also have higher expectation in the applications they interact with, such as immediate responses and smooth experience in their interaction. Thus, the need for a better mechanism to handle massive amount of data in short period of time and meet real-time constraints is a must in nowadays computing system, and one of the solution is In-memory Computing (IMC). IMC refers to a technique that, instead of using complex data format that stored in slowed access disk storage, the data now resides directly in memory for quick random access. The random access speed to memory is about thousand times faster than mechanical hard drive. The gap is so large, that not only using IMC solves the problem of speeding up processing time, it opens many other computational areas that using mechanical hard disk storage just cannot afford.

Following the trend of IMC, one particular computing service we look at is the Virtual Desktop Service (VDS), that normally runs on top of a Virtual Desktop Infrastructure (VDI) as illustrated in Figure 1. Nowadays, an enterprise VDI system is analogous to a cloud platform, that contains multiple physical servers run on virtualization hypervisor such as Xen [2] or VMware vSphere [3]. Then, virtual desktop machines (VDMs) can be created by these virtualization platforms. Each VDM runs a complete desktop environment, its resources including Central-Processing-Unit (CPU), memory and hard-drive are virtualized from physical hardware, this makes resource scaling and provisioning simpler. End-users connect to those VDMs remotely using thin client terminal software utilizes different protocols such as Virtual Network Computing (VNC) or Remote Desktop Protocol (RDP). We call this environment Virtual Desktop Environment (VDE). Even-thought traditional VDE offers sufficient performance to normal users, power users who need fast computational machine and smoothness in their interactions still find traditional VDE does not give them the satisfactory. And usually the bottle neck comes from the slow access speed of physical hard drive. It is much slower in I/O access than memory flash such as DRAM (Dynamic Random Access Memory). In this paper, we experiment a different approach to provide the VDE whereas the whole environment stays in RAM, and we call this new VDE approach the ”In-Memory Virtual Desktop Environment” (IM-VDE).

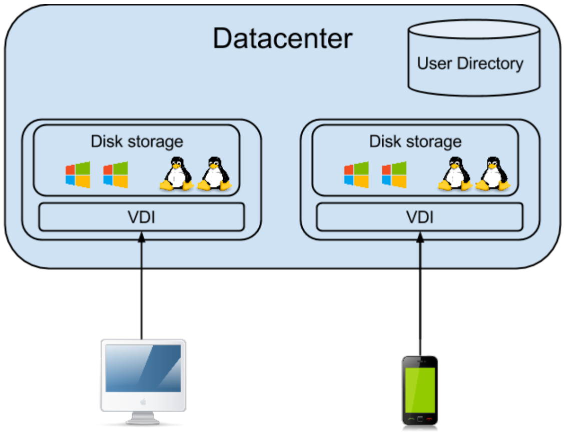

Fig. 1: A typical VDI environment

The remaining of the paper is organized as follows: chapter two shows related work within IMC field, chapter three shows our approach of the experiments. Chapter four elaborates the results and the issues we encountered within our experiment on IM-VDE, what are the drawbacks that need to be resolved, and we draw our conclusion in chapter five.

II. Related Work

In enterprise environment, many business fields require real-time processing speed such as banking, trading platforms, large e-commerce shops, those that have thousand of requests per second. These workload need to be processed as fast as possible, as most often the bottleneck that slowing down the processing time is the I/O access of disk storage. To overcome this, IMC approach is being adopted in many computing areas.

Test	Description
CPU Blowfish	Test CPU using strong encryption algorithm with symmetric block cipher that can be used to replace DES or IDEA.
CPU CryptoHash	Test CPU using hash function that takes an arbitrary block of data and returns a string with fixed-size in bit unit.
CPU Fibonacci	Test CPU with the Fibonacci function using recursive coding and parallel programming.
CPU N-Queens	Test CPU with N-Queen chess algorithm, with optimal complexity is N factorial for brute-force approach.
FPU FFT	Test CPU with Floating-Point Unit-Fast Fourier transform algorithm.
FPU Raytracing	Test CPU with Floating-Point Uni for ray tracing problem in computer graphic.
MySql test	Transactional test my MySql database, including both I/O access and CPU performance.
Disk I/O test	Pure I/O access test, benchmarking of disk speed in reading and writing data.
Video smoothness test	Client-side perspective test, determine by the smoothness of video frame-rate.

TABLE I: Benchmarking tests' detail

Zhiqiang Ma et.al. [4] proposed VOLUME as a large-scale in-memory general solution for distributed file system. Following MapReduce programming model [5], VOLUME takes both memory and disk storage to become one unified memory environment. Their new IMC approach help developers avoid worrying about data persistent on disk, simplify the programming work-flow, and significantly increase the system's performance. Their prototype shows impressive result in compare to Hadoop [6] and Spark [7], other popular open-source distributions of MapReduce model.

In [8], the authors try to tackle big-data analysis problem by combining parallel computing with in-memory technique. Their framework utilizes the parallel processing power of multiple graphics-processing-units (GPU), and performs in-memory text data analysis with their custom architecture. The simulation results showed significant improvement in performance.

Another interesting research topic of IMC is detailed in [9], in which the authors built a Version Control System (VCS) that runs in-memory in the cloud, called Protego. Their system design approach is considered to be flexible, automation with scalable characteristic of the cloud environment.

III. In-Memory VDE approach

A. *Methodology*

We conduct the performance benchmark based on two principals, server-side performance and client-side user-perceived performance. A set of synthetic benchmark tests are used to analyses the server-side performance based on CPU load, I/O access speed and overall running speed of the VDMs. On the other hand, the testing for the end-user perspective is conducted as a survey by a group of volunteers, who will in turn watch a recorded video showing graphical testing results of normal and IM VDM, then submit their score to the survey. Table I details all the benchmarking tests.

First, we need to simulate an IM-VDE environment in our commodity research hardware, and the technique we used usually being called as "RAM disk". It is a method allows the OS to allocate a block of RAM and treat that block as if it was a normal disk storage block. After that, we are allowed to put any arbitrary data in and use normal operations of hard-drive storage on that RAMdisk region, but still have the benefit of I/O access speed of DRAM. Hence, by duplicating one VDM and put it into the RAMdisk, we essentially achieve the result of having an In-memory Virtual Desktop Machine (IM-VDM), or in other word, a simple IM-VDE.

In fact, RAMdisk has been a popular technique used in many Linus distributions. For example: Puppy Linux is one special Linux distribution that when loaded, the whole OS is resided in memory. Thus, it's application execution times is extremely fast. There are several RAMdisk file-systems, but the two most often used are RamFS and TmpFS. RamFS is a simple file-system that exports Linux's disk caching mechanisms as a dynamically re-sizable ram-based file-system. TmpFS is similar to RamFS, but its size can be fixed, and it uses swap space in case of low memory situation. Based on [10], the authors claimed that TmpFS gives better performance for in-memory file-system running on non-volatile memory. As such, we have decided to use TmpFS to simulate our in-memory file-system for the IM-VDE testbed.

With the testbed ready, the second step in our research is to evaluate the performance differences between IM-VDM and non IM-VDM. Benchmarking of the VDE can be challenging since there are no clear methods to measure the performance from the client-side perspective, since it depends on many factors, which can be the network status, or the remote control protocol being used, or most importantly, the end-user experience. But there are some excellent researches that layout novel methodology to benchmark the VDE. For example, in [11], Alex Berryman et.al. introduced a benchmarking toolkit for thin-client in VDE, whereas they used a technique call "slow-motion benchmarking" to measure the transmission time, network bandwidth and quality of streaming images. Our experiments utilize some of the techniques that they introduced.

B. *Server-side performance benchmark*

In preparation for the benchmarking, we use a physical server with hardware specifications as follow: Intel(R) Xeon(R) CPU E3-1220 V2, 3.10GHz with 4 cores, 32 gigabytes of DRAM memory, one SSD, one mechanical hard drive (HD)and the host OS is Linux distribution "Ubuntu Saucy 13.10". Next step, we prepare three VDMs using "VMware Player" software with the guest OS is also "Ubuntu Saucy 13.10". We then distribute those VDMs to reside on different file-systems, one stays in SSD, one stays in HD and the last one resides in TmpFS, Table II gives the details of testbed setup. Once the set-up is done, we conduct our benchmarking using state-of-the-art open-source software to accomplish this task. The first task is using a benchmark tool call "HardInfo: system profiler and benchmark" [12]. "HardInfo" can gather information about the OS environment, and do several benchmarks that mostly related to CPU-bound performance. By performing each test several time, we take the average results and compare the differences. The benchmark's result is presented in Figure 2. Beside "HardInfo", we use another open-source tool called "SysBench", which is a modular, cross-platform and multi-threaded benchmark tool. One particular salient feature of this tool is the focus on database benchmark under intensive load. This will give us some perspectives in real life practical

Name	Configuration
Host OS	Intel(R) Xeon(R) CPU E3-1220 V2, 3.10GHz with 4 cores, 32 GB DRAM, runs Ubuntu 13.10
HD VM	1vCPU, 1 GB RAM, 20GB disk storage, runs Ubuntu 13.10, RealVNC protocol
SSD VM	1vCPU, 1 GB RAM, 20GB disk storage, runs Ubuntu 13.10, RealVNC protocol
TmpFS VM	1vCPU, 1 GB RAM, 20GB disk storage, runs Ubuntu 13.10, RealVNC protocol

TABLE II: Testbed details

environment. Similar to aforementioned benchmarking, we calculate the average results of several tests and compare with other VDMs. The benchmarking result is illustrated in Figure 3.

C. User-perceived performance benchmark

Benchmarking the VDE performance from the end-user perspective is not a simple task. It does not work by simply running a few benchmark software to know the results. In the VDE, user interaction with the remote graphical terminal is accomplished by using different remote access protocols such as VNC or RDP, which are commonly used in Linux and Windows OS respectively. Hence, to measure the performance of VDMs when users are using them, we mainly concern how quick and smoothness the rendering of remote server's graphical environment to client terminal. Another factor to consider is the type of end-user who operates the client. An office worker who mainly works with word processing application can tolerate the performance of VDM better than a power user such as a programmer or designer, where the smoothness and continuous flow of interactive responses are very important to them. Lastly, the network condition also plays a crucial role in the VDE, but in our test, we assume that the network condition is stable, since what we focus on is how the performance of a VDM running in-memory differs with a VDM running on HD. Thus with a stable network, it can help us to see the differences better.

With these regards, we have decided to conduct the tests as follows: we use the VNC protocol for the remote graphical solution, since it is open-source and platform independent. Then, for each VDM on each file-system, we respectively remotely connect to the machine, run automate script to start up many popular applications sequentially, and finally play the classical ”Big Buck Bunny” video. We then record a video of the interactive session on the client-side. After that, we combine those recorded videos by putting into a side-by-side comparison video-clip , and gives it to a group of volunteers who will then rate them in scale from one to ten. By using this method, we can really reflect the end-user perspective on how they perceive the performance. The survey result is shown in Figure 4.

IV. Evaluation

In this section, we give our evaluation of the benchmarking results regarding the experiments we conducted, and discuss some of the challenges, problems we found during the testing.

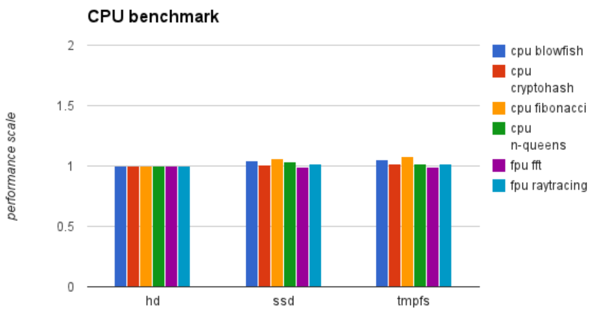

Fig. 2: HardInfo benchmark

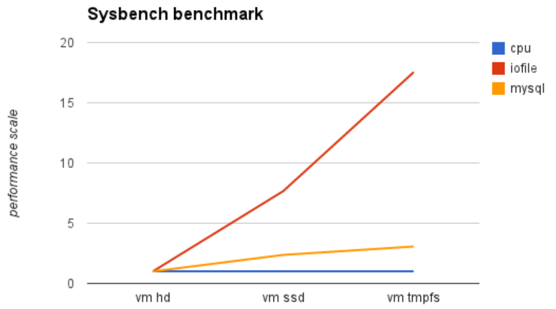

Fig. 3: Sysbench benchmark

A. Results

From Figure 2, we conclude that in term of CPU-bound benchmark, the IM-VDM has the same performance as with normal hard disk VDM (HD-VDM) and SSD-VDM. It is understandable since the synthetic benchmarks of the CPU do not rely on other hardware components such as memory and/or persistent device. And given VDMs with identical CPUs, they should yield similar performing results. But when we look at Figure 3, a big gap between IM-VDM and HD-VDM is clearly seen. The I/O access speed gives a huge speed up, thus this is one of the main benefit we get from IM-VDM, that is the access speed of data. Not only it speeds up the processing power of the applications, it also provides a better computational mechanism to solve problems faster.

For the client-side, the survey result in Figure 4 shows some differences in user-perceived performance. During the process of booting up multiple applications, thanks to speedy I/O access, applications start almost instantly, therefore we have high-score ratings from the volunteers. On the other hand, for the streaming of ”Big buck bunny” video, which is a CPU-bound task, shows equally average results.

B. Challenges

Even though utilizing VDM to run in-memory give us positive boost in performance, there are challenges and issues we encountered that should be addressed. Firstly, as soon as we

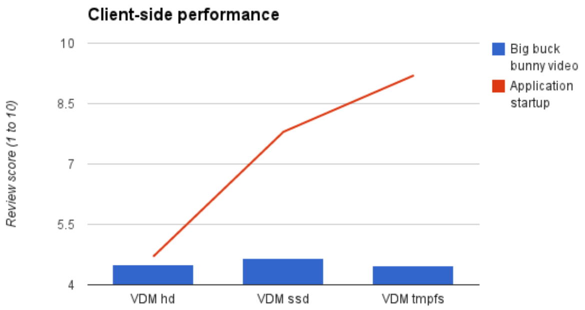

Fig. 4: Survey result

start the benchmarking, we have disk space problem with IM-VDM. Since we have limite amount of DRAM in our testbed, the IM-VDM cannot affort to have large storage size. This is the common issue for IMC approach. To overcome this, we need to provisioning lot of DRAM memory, this leads the another problem, which is cost effective. We know the price for DRAM memory has been reduced significantly over years, but the price of HD also reduces a lot. Given the price per gigabytes of nowadays commodity hardware, a DRAM flash for desktop would cost $10/GB, while HD's cost is only $0.08/GB.

Not only we have disadvantage in storage space, using TmpFS also wastes lot of memory resources, as we encountered the phenomenon we called ”double memory usage”. Because the OS treats the TmpFS memory block as normal disk block, it still loads the data into main memory one more time to process that data instead of accessing the memory block directly in the TmpFS region. This in the worse case will cost twice the amount of memory usage for a particular VDM.

Last but not least, we have fault-tolerance and persistent issues. Since DRAM is volatile, in case of unwanted events happen such as electricity lost, all the VDMs will lost its data. Hence, we need a proper backup mechanism for in-memory file-system, as also mentioned in [10]

V. Conclusion

After thorough benchmarking of our IM-VDE testbed, it is clear that IM-VDE provides positive benchmark results. But there are many concerns that we need to take into consideration when using IM-VDE. The first one is the performance-gain in trade-off with the storage shortage because the cost of DRAM is still quite high in comparison with HD, but it provides significant boost up in term of computing power. The second concern is the type of end-user for the VDE. If he/she runs tasks that depend on CPU but not much I/O accesses involve, the positive effects are often unnoticeable. Currently, IM-VDE is still a relativity new Virtual Desktop Environment paradigm, there are still many challenges but we believe IM-VDE will set the new trend.

Acknowledgment

This research was supported by the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2013- H0301-13-4006) supervised by the NIPA(National IT Industry Promotion Agency). Dr. CS Hong is the corresponding author.

References

[1] ”Harness the Power of SAP HANA In-Memory Computing.” In-Memory Technology. http://www.sap.com/pc/tech/in-memory-computing-hana.html

[2] The Xen Project, the Powerful Open Source Industry Standard for Virtualization.” The Xen Project, the Powerful Open Source Industry Standard for Virtualization. http://www.xenproject.org

[3] ”VSphere Hypervisor.” Free VMware , Free Virtualization (ESXi). http://www.vmware.com/products/vsphere-hypervisor

[4] Ma, Zhiqiang; Hong, Ke; Gu, Lin, ”VOLUME: Enable Large-Scale In-Memory Computation on Commodity Clusters,” Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on , vol.1, no., pp.56,63, 2-5 Dec. 2013

[5] Ranger, C.; Raghuraman, R.; Penmetsa, A.; Bradski, G.; Kozyrakis, C., ”Evaluating MapReduce for Multi-core and Multiprocessor Systems,” High Performance Computer Architecture, 2007. HPCA 2007. IEEE 13th International Symposium on , vol., no., pp.13,24, 10-14 Feb. 2007

[6] Shvachko, K.; Hairong Kuang; Radia, S.; Chansler, R., ”The Hadoop Distributed File System,” Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on , vol., no., pp.1,10, 3-7 May 2010

[7] Karau, Holden. Fast Data Processing With Spark. Packt Publishing Ltd, 2013.

[8] Poh Kit Chong; Karuppiah, E.K.; Keh Kok Yong, ”A Multi-GPU Framework for In-Memory Text Data Analytics,” Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on , vol., no., pp.1411,1416, 25-28 March 2013

[9] Gioachin, F.; Qianhui Liang; Yuxia Yao; Bu-Sung Lee, ”Protego: In-Memory Version Control System in the Cloud,” Software Engineering Conference (APSEC), 2012 19th Asia-Pacific , vol.1, no., pp.232,239, 4-7 Dec. 2012

[10] Hyunjun Kim, Joonwook Ahn, Sungtae Ryu, Jungsik Choi, and Hwansoo Han. 2013. In-memory file system for non-volatile memory. In Proceedings of the 2013 Research in Adaptive and Convergent Systems (RACS '13). ACM, New York, NY, USA, 479-484.

[11] Berryman, A.; Calyam, P.; Honigford, M.; Lai, A.M., ”VDBench: A Benchmarking Toolkit for Thin-Client Based Virtual Desktop Environments,” Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on , vol., no., pp.480,487, Nov. 30 2010-Dec. 3 2010

[12] ”HardInfo.” : System Profiler and Benchmark. http://hardinfo.berlios.de