International Journal of Advanced Smart Convergence Vol.9 No.2 114-122 (2020)
http://dx.doi.org/10.7236/IJASC.2020.9.2.114

IJASC 20-2-13

A Performance Analysis of Virtualization using Docker for Radar Signal Processing

Jong-Hoon Ji, Hyun-Wook Moon, Sung-Hwan Sohn, Sung-Min Hong, Se-Woong Kwon, Yeon-Duk Kang

Research Engineer, Radar R&D, LIG Nex1, Korea
E-mail: jonghoon.ji@lignex1.com

Abstract

When replacing hardware due to obsolescence, discontinuation, and expansion of software-equipped electronic equipment, software changes are required in the past, but if virtualization technology is applied, it can be applied without software changes. In this regard, we studied in order to apply virtualization technology in the development of naval multi-function radar signal processing, we studied hardware and OS independency for Docker and performance comparison between Docker and virtual machine. As a result, it was confirmed that hardware and OS independence exist when using Docker and that high-speed processing is possible compared to the virtual machine.

Keywords: *Docker, Maintainability, MFR(Multi-Function Radar), Scalability, Virtualization, Virtual Machine.*

1. Introduction

Recently, when developing a radar for ships, MFR(Multi-Function Radar) has been developed to perform various missions and make efforts to prepare for the naval survivability and future battlefield environment[1][2].

MFR(Multi-Function Radar) for these ships can also be detected by scanning multiple areas through electronic beam steering, and by searching and tracking multiple targets to increase survivability. Operational maneuverability can be improved and for this reason a significant contribution to the increase in naval defense is possible[3][4].

However, during the development and production of such a MFR(Multi-Function Radar) for ships, a lot of time is required, and due to the rapid development speed of electronic equipment and software, software aging and OS upgrade may be required. In addition, it may be necessary to expand computer resources in response to changes in the military tactical environment. That is, there is a case where the replacement of hardware or OS is required depending on the aging, discontinuation, and expansion of hardware and OS

Manuscript Received: April. 15, 2020 / Revised: April. 20, 2020 / Accepted: April. 24, 2020
Corresponding Author: jonghoon.ji@lignex1.com
Tel: +82-31-288-9767, Fax: +82-31-288-9122
Research Engineer, Radar R&D, LIG Nex1, Korea

regardless of the improvement of software performance. The problem is that the software is dependent on the hardware or OS, so when the hardware or OS changes, the software also needs improvement. This can cause software defects in terms of maintenance of the weapon system. As shown in Figure 1, hardware has high defects in the initial manufacturing process, and the defect rate decreases until the end of the hardware life, and when the durability decreases, the defect rate increases again, and the graph appears in the form of a 'Bathtub curve' while the software is physical or environmental. Although it is not affected, the defect is greatly increased when the software is changed, and the same applies to the development of a radar weapon system[5].

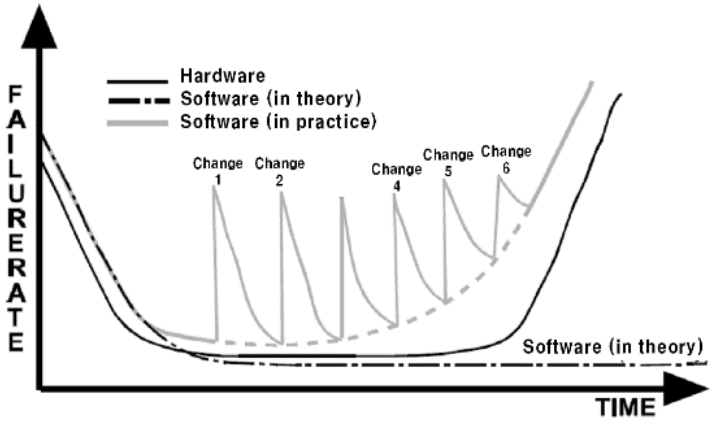

Figure 1. Change of Failure rates by software and hardware maintenance

Meanwhile, cases of research and weapon system development using virtualization technology are increasing[9][10]. Virtualization technology has the advantage of efficiently using computer resources to reduce the total cost of ownership and ensure independent operation between virtual machines, and it is possible to apply software independent from hardware and OS because scalability increases due to ease of integration of computer resources. According to these advantages, recent research cases focus on virtualization technology for data center and server operation[11], but there are few research and development cases for virtualization technology in areas requiring large-scale high-speed data processing.

In this paper, when developing a signal processing system that requires large capacity and high-speed data processing in a MFR(Multi Function Radar) for ships, virtualization technology is applied to increase maintenance and to expand computer resources when hardware and OS replacement is required. Given that it is possible to flexibly cope with the problem, performance analysis according to hardware and OS independence and virtualization technology was performed by applying virtualization technology to the signal processing system.

2. Virtualization technology

Virtualization technology has emerged to provide users with a virtualized operating environment. Virtualized operating environments have VM(Virtual Machines) and Container.

VM(Virtual Machine) and Container technology have their pros and cons, as shown in Table 1, and despite the increasing interest in containers, they are still used in most cloud environments. In the case of container virtualization, since the processing layer is less than that of the VM(Virtual Machine), as shown in Figure 2, Since the Container does not have a Guest OS, it uses less computing resources and has a faster

processing speed. In the case of the VM(Virtual Machine), OS virtualization is applicable as the kernel resources are separable. That is, by applying virtualization software, process cores and memory can be directly allocated from individual OS.

Table 1. Container vs Virtual Machine

	Container	Virtual Machine
Hypervisor	Non-Existence …	Existence
Guest OS	Non-Existence	Existence
Kernel Resource Separation	Impossible	Possible
Processing Time	Fast	Slow
Resource efficiency	High …	Low

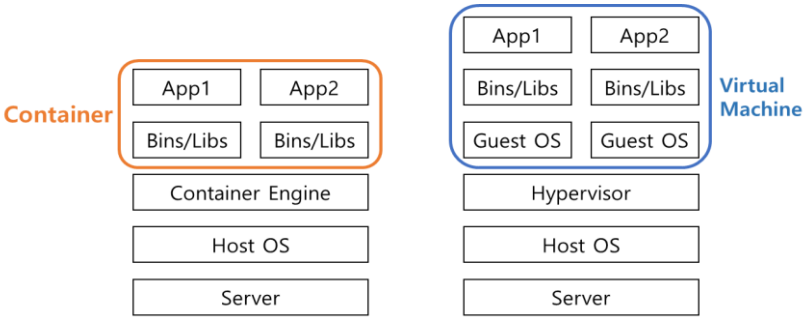

Figure 2. Container vs Virtual Machine

2.1 Virtual Machine(VM)

The VM(Virtual Machine) virtualizes the entire hardware stack using a hypervisor and provides it to the user. Hypervisor, the core technology of virtual machine use, is classified into Type 1 and Type 2 according to location and role.

In the case of the Type 1 hypervisor, it is installed and operated directly on the host hardware, and the Type 2 hypervisor is installed on the host OS. Although there are differences in the location of the hypervisors, VM(Virtual Machine) have relatively large overhead because resources are commonly allocated from hardware through a hardware emulator called hypervisor. In addition, a separate guest OS installation is required in addition to the host OS, and computer resources beyond the process resources executed for the operation of the guest OS are used. However, the VM(Virtual Machine) also has an advantage in that it is possible to use a separate guest OS kernel from the host OS by separating kernel resources[6][8].

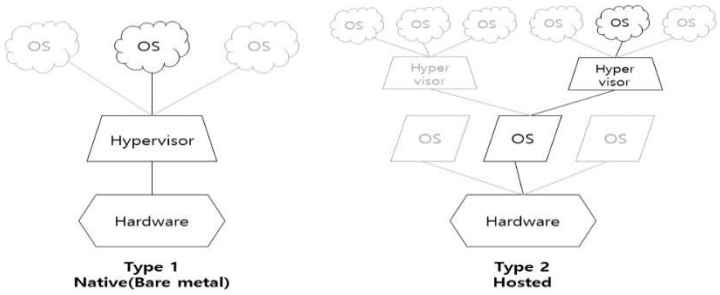

Figure 3. Hypervisor Type

2.2 Container

Container is an OS-level virtualization technology that shares the Linux kernel and executes processes in an isolated environment. Containerized virtualization technology is characterized by being lighter and more portable than VM(Virtual Machines). The Linux container's OS is divided into a 'kernel space' for managing physical resources and a 'user space' for executing user processes(applications). Containerized virtualization refers to limiting the resources available to each user process by dividing the user space into multiple groups[6] as shown in Figure 4. In this way, the space separated by arranging several user processes is called a 'Container'.

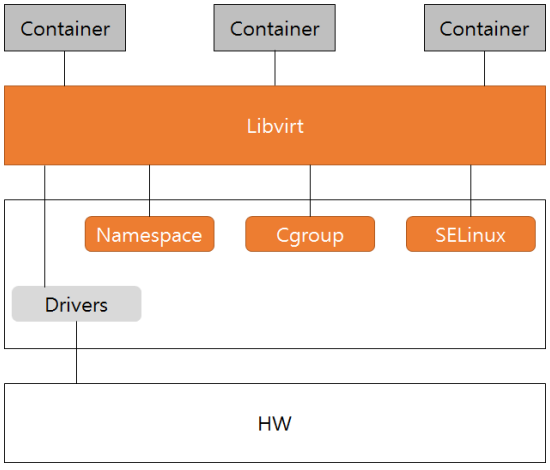

Figure 4. Container Architect

Containerized virtualization technology has high speed, efficiency, and high portability because there is no separate hardware emulator. All containers have a separate execution environment, not a host environment, which is composed of files and can be shared in an image format. If you use the Linux kernel, you can share and reproduce the container's execution environment. In addition, since the environment in which the container runs is independent, it does not affect other containers[8]. A representative Docker exists in such container technology.

3. Test environment

In this study, the following test environment was constructed to compare hardware and OS independence for Docker and performance based on virtualization technology.

Table 2. Hardware Specification

Spec	Workstation	Desktop PC
CPU	Intel(R) Xeon(R) CPU E5-2648L v3 1.80GHz 48Core …	Intel(R) Core(TM) i7-7280HQ CPU 2.90GHz 4Core
Memory	48GB …	8GB

3.1 Hardware and OS independence test environment for Docker

First, hardware independence test for Docker, the same OS (Linux CentOS6.6) and Docker engine 1.7.1

were installed on workstations and desktop PC with different hardware specifications as shown in Table 2, and one of the radar signal processing algorithms, Pulse Compression was repeated 500 times. Pulse compression is an algorithm that generates a reference signal and performs convolution between an input signal and a reference signal. The actual implementation performs a FFT(Fast Fourier Transform) on each signal, multiplies the two signals, and performs an IFFT(Inverse Fast Fourier Transform) to calculate the corresponding computation time. The actual implementation is shown in Figure.5

$$y(t) = x(t) * x_{ref}(t) \tag{1}$$

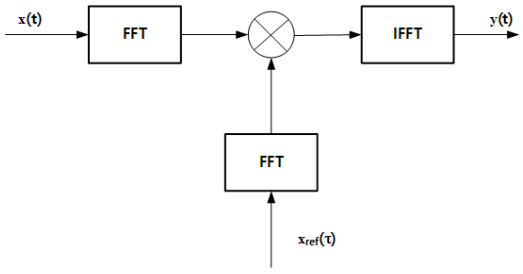

Figure 5. Fast Fourier Transform based Pulse Compression Processing Block Diagram

3.2 Performance analysis test environment according to virtualization technology

In order to compare and analyze the performance difference according to the virtualization technology, after installing the same workstation and OS (Linux CentOS6.6), Docker Engine 1.7.1 and VMware 15.5 were respectively installed to perform pulse compression 500 times to measure the calculation time.

4. Test results and analysis

In this chapter, we derived and analyzed the results of Docker's hardware and OS independence and performance comparison with VM(Virtual Machine).

4.1 Hardware independence test results for Docker

When applying a Docker, developing a radar signal processing system, install the same OS on different hardware workstations and desktop PC in Table 2, and check whether Docker is installed or not, as described above, to confirm that maintenance is maintained by hardware independence. The resulting pulse compression calculation time was measured. In the absence of a Docker, pulse compression is performed by the processing power of the actual hardware, which is assumed to be a real machine, and compared to the case where a Docker is present. The average of the calculation time measured after 500 repetition tests was obtained, and the performance degradation rate (ρ) by the Docker was calculated as follows.

$$\rho = 100 - \frac{T_{RM}}{T_{Docker}} \times 100[\%] \tag{2}$$

Here, T_{RM} is the average calculation time measured on the RM(Real Machine), and T_{Docker} is the average calculation time measured on the Docker. In addition, assuming large capacity, high-speed data processing, the computational time was measured for computer loads of 0%, 25%, and 50% to derive performance differences. Figure 6 shows the result of deriving the average calculation time according to the computer load on RM(Real Machine) and Dockers on workstations and desktop PC, and Table 3 summarizes the values.

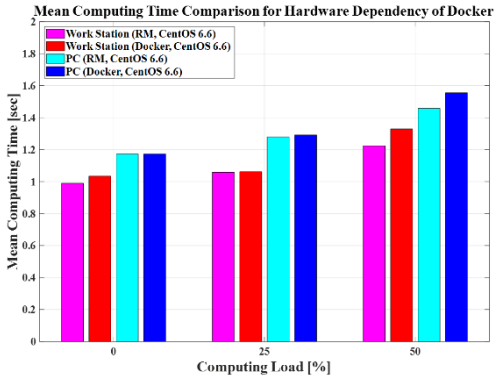

Figure 6. Mean computing time of Real Machine and Docker according to the hardware specification with computing load

Table 3. Comparison of mean computing time and performance degradation rate between Real Machine and Docker according to the hardware specification with computing load

		Computing Load		
		0%	25%	50%
Workstation	Real Machine	1.092 …	1.139	1.224
	Docker	1.103	1.142	1.33
	Performance degradation rate	1%	0.3%	7.97%
Desktop PC	Real Machine	1.172	1.28	1.458
	Docker	1.173	1.292	1.556
	Performance degradation rate	0.001%	0.93%	6.3%

Looking at the results, the average processing time varies depending on the specifications of the hardware, but the hardware independence can be confirmed because it has almost the same processing time compared to the RM(Real Machine) and Docker. However, when the computer load is 25% or less, the performance degradation rate is around 1%, but when the computer load increases to 50% or more, it is confirmed that the performance decreases slightly to 7.97% on the workstation and 6.3% on the desktop PC. In other words, theoretically, when applying virtualization technology, it must be hardware-independent, but when the computer load is high, about 10% of performance degradation may occur, so it is necessary to design a computer resource in consideration of this.

4.2 OS independence test results for Docker

When applying the Docker, developing a radar signal processing system, install the different operating systems, Linux CentOS6.6 and Linux CentOS7.7, on the desktop PC, and measure the pulse compression calculation time according to the presence or absence of Docker installation.

Figure 7 shows the result of deriving the average calculation time according to the computer load of the real machine and Docker in different OS, and Table 4 shows the table. Looking at the results, although the average processing time varies depending on the OS version, OS independence can be confirmed because it has almost the same performance compared to the RM(Real Machine) and Docker. However, when the computer load is 25% or less, the performance degradation rate is around 1%, but when the computer load is 50% or more, it can be seen that it decreases to 6.3% in Linux CentOS6.6 and 5.7% in Linux CentOS7.7. As in the trend in A. above, if the computer load on the hardware is large, some performance degradation may

occur.

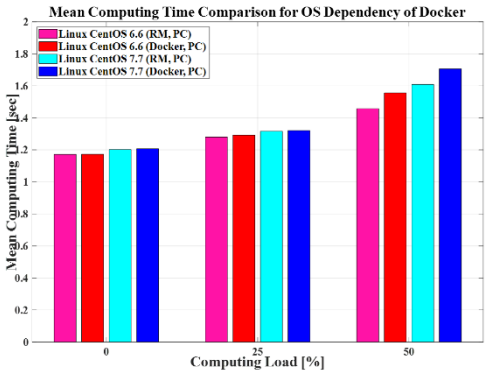

Figure 7. Mean computing time of Real Machine and Docker according to the OS Version with computing load

Table 4. Comparison of mean computing time and performance degradation rate between Real Machine and Docker according to the OS version with computing load

		Computing Load		
		0%	25%	50%
Linux CentOS 6.6	Real Machine	1.172	1.28	1.458
	Docker	1.173	1.292	1.556
	Performance degradation rate	0.01%	0.93%	6.3%
Linux CentOS 7.7	Real Machine	1.202	1.316	1.609
	Docker	1.207	1.321	1.706
	Performance degradation rate	0.04%	0.04%	5.7%

4.3 Performance comparison analysis result according to virtualization technology

When developing a radar signal processing system, after installing Linux CentOS6.6 on a workstation to check processing performance according to virtualization technology, pulse compression calculation time according to VM(Virtual Machine) and Docker installation was measured.

Figure 8 shows the results of slaughtering the average calculation time according to the computer load of RM(Real Machine), VM(Virtual Machine), and Docker, and Table 5 summarizes the values. Looking at the results, Docker is almost the same as a real machine when the computer load is 25% or less, but when the computer load is 50% or more, the performance decreases by around 10%. On the other hand, in the case of the virtual machine, the performance was lowered more than the Docker, and when the computer load increased, the performance degradation occurred by about 79%. In the case of a VM(Virtual Machine), there is a CPU steal time, which is a time that the virtual CPU waits to receive real CPU resources. As the computer load of the host kernel increases, the CPU steal time of the virtual machine increases and the processing time increases. That is, as more resources are processed in the host kernel, the virtual machine does not receive hardware resources and does not perform processing, so the processing time increases. Therefore, Docker is more suitable than a virtual machine to apply virtualization techniques to large-capacity, high-speed data processing.

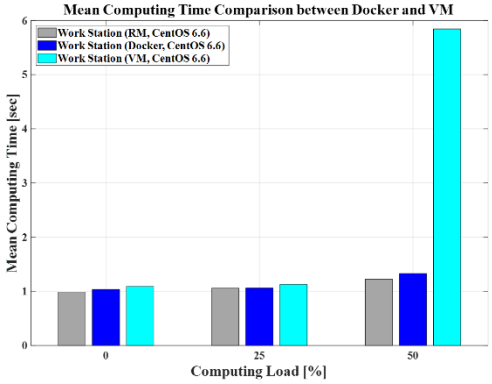

Figure 8. Mean computing time of Real Machine, Virtual Machine and Docker according to the computing load

Table 5. Comparison of mean computing time and performance degradation rate of Real Machine, Virtual Machine, and Docker according to the computing load

	Computing Load		
	0%	25%	50%
Real Machine	1.092	1.139	1.224
Docker	1.103	1.142	1.33
Performance degradation rate	1%	0.3%	7.97%
Virtual Machine	1.146	1.248	5.845
Performance degradation rate	4.7%	8.7%	7.91%

5. Conclusion

When hardware changes such as aging, discontinued, and need to expand software-equipped electronic equipment are required, software changes must be carried out in the past, but if virtualization technology is applied, it can be applied without software changes, which is advantageous in terms of maintainability and scalability. Moreover, Docker has great advantages in portability. In general, application, library, and kernel environment settings are required to build equipment. Especially in the case of environment setting, if there is no manual, there is a high possibility of making a mistake and it can cause a failure. Docker can easily be built into other equipment through storage devices, including USB, by imaging all environments for driving applications. The configuration file is also included in the image, so it can be easily and quickly ported. Therefore, in this paper, hardware and OS independence and performance comparison with another virtualization technology VM(Virtual Machine) were performed for Docker to apply virtualization technology when developing a signal processing system for MFR(Multi-Function Radar) for ships. As a result, it was confirmed that it has independence for hardware and OS, and it was found that it is advantageous in terms of maintenance. However, when the computer load is large, performance degradation of around 10% occurs, so it is necessary to design a computer resource considering this. Also, as a result of comparing the performance with the VM(Virtual Machine), it was confirmed that the VM(Virtual Machine) is not suitable for high-speed and high-speed data processing, while Docker maintains more than 90% performance compared to the RM(Real Machine) when applied, making it more suitable for high-speed and high-speed data processing. Therefore, it is expected that if the Docker is applied when developing a MFR(Multi Function Radar) for ships in the future, maintenance and scalability can be obtained.

References

[1] Kim Jeong-ryul, “Technical status and development trend of multi-function radar for ships 1”, Defense and technology 367, pp 38-47, Sep 2009

[2] Kim Jeong-ryul, “Technical status and development trend of multi-function radar for ships 2”, Defense and technology 368, pp 62-73, Oct 2009.

[3] Yang Jae-chang, “Development trends of multi-function radar for ships around the world 1”, Defense and technology 469, pp 96-111, Mar 2018.

[4] Yang Jae-chang, “Development trends of multi-function radar for ships around the world 2”, Defense and technology 470, pp 120-141, Apr 2018.

[5] Kwan-Young Lee, Sang-Moon Kim, Eun-Shim Park, Jae-Eun Park, Geun-Hyung Kim, “Qualitative study of software ILS application: Comparison analysis of maintenance types in software and hardware”, Jounal of the Korea Academia-Industrial cooperation Society Vol. 15 No. 9 pp. 5726-5737, pp 120-141, 2014.

[6] Jae-Hong Lee, Docker for the Really Impatient, Gilbut, 2014.

[7] Jung-Yeon Hwang, HoYong Ryu, “Performance Comparison and Forecast Analysis between KVM and Docker”, The Journal of Korean Institute of Information Technology, Vol. 13, No. 11, pp. 127-136, Nov 2015

[8] David Clinton, Linux in Action, Gilbut, 2019

[9] Sang-min Kwon, Seung-mo Jung, “Virtualization based high efficiency naval combat management system design and performance analysis”, Journal of the Korea Society of Computer and Information, Vol.23, No. 11, pp. 9-15, Nov 2018.

[10] Jin-Yong Im, Dong-Seong Kim, “Performance Evaluation of Commercial Virtualization Scheme for Next Generation Naval Combat System”, Proceeding of Symposium of the Korean Institute of communications and Information Sciences, pp. 368-369, Jan 2016.

[11] Hwan-tae Kim, Hwang-nam Kim “Control Algorithm for Virtual Machine-Level Fairness in Virtualized Cloud Data Center”, The Journal of Korean Institute of Communications and Information Sciences, pp. 512-520, Jan 2013.