# Comparison of Windows and Linux as Docker Hosts

Sebastian Pfaller
Student, Eastern Institute of Technology
Pfalls1@student.eit.ac.nz

Thomas Hartley
Senior Lecturer, EIT
thartley@eit.ac.nz

## ABSTRACT

Containerization is a new approach to virtualization in software development and deployment. Docker is the most popular containerization solution and was adopted by Linux in 2013. Windows announced in 2014 that it was working towards including the popular Docker engine into its next Server Release. In June of 2016, Microsoft publicized that it had integrated the Docker container engine into Windows 10 and Server 2016 with Hyper-V Containers. A comparative study of the installation methods for both Windows and Linux are examined in this paper. While Linux is an Open Source Operating System which allows developers to examine the source code including the Docker engine, Windows is not. This created a challenge in the literature research for this paper, while also offering opportunities to share findings with the wider educational community.

**Keywords**: Windows, Linux, Docker, Containerization, Hosts, Limitations

## 1. INTRODUCTION

Containerization is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it including the code, runtime, system tools, system libraries, settings (Docker Inc, 2017a). They are available for both Linux and Windows based applications and its use means that the software which runs within these environments will always run the same, regardless of the environment.

Its application within education is limited but growing, as it offers a way to overcome the limitations of limited number of computers available in labs and hours and opportunities for students to cover a wide variety of set of practical exercises that cover the use of complex information systems (Cheremisina, Belov, Tokareva, Nabiullin, Grishko & Sorokin, 2017). To support the learning of these multicomponent information systems would access to powerful hardware requirements which would be difficult to provide and support.

Docker is a platform which provides this container environment. Docker uses a client-server architecture in which the Docker-client interacts with the Docker daemon, enabling the operations of creating and launching containers on the server. This approach would allow for safe sandboxing where by any application running on the OS will not compromising the base operating system, therefore allowing the student to carry out any experiments but still maintaining the highest level of performance (Cheremisina, et. al,. 2017).

Despite these advantages, containerization and Docker is relatively new. Docker is a relatively new technology first released in 2013 (Tozzi, 2017). However is one of the most commonly used solution for containerization (Statista Inc., 2016), with a yearly growth rates of up to 40% according to Datadog Inc. (2017). Its greatest advantage is that it offers support for both Linux and Windows platforms, however these platforms offer different constraints and advantages, therefore, understanding the implications of using different platforms is important when considering which software stack you would like to use or when deciding to implement both within a software solution.

This quality assured paper appeared at the 9th annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ2018) and the 31st Annual Conference of the National Advisory Committee on Computing Qualifications, Wellington, NZ, July 11-13, 2018 as part of ITx 2018.

Within the literature most studies generally only compare the performance of different host systems for Docker, in this study it will rather focus on the differences between the different software stacks and the implications of each. The aim is to provide insight to institutes and business with a clear comparison between these two environments when using Docker on the two guest OS's.

In this research paper, we will address the specific research question of:

- What are the differences between running Docker on a Linux and on a Windows host?

The structure of the paper is as follows; firstly, the paper will provide an overview of the existing literature relating to containerization and Docker, next in section **Error! Reference source not found.** we provide an overview on how we evaluated the two systems and the methodology we adopted to do this. In section 3, 4 and 6 we provide our findings of this experiment, a discussion and the implications of these findings. Lastly, we concluded our study, outline the limitations and provide an outlook on possible, future investigations.

## 2. LITERATURE REVIEW

Until now the meaning of virtualization implied the hypervisor-based approach however this has since been expanded to include containerization. In Eder (2016), he provides an overview of the differences between hypervisor-based as well as container-based virtualization. The differences are, "Hypervisor-based virtualization provides strong isolation of a complete operating system whereas container-based virtualization strives to isolate processes from other processes at little resource costs".

In Eder (2016) the concept of Linux Containers (LXC) is explained in detail and highlights the necessary components like kernel namespaces, cgroups and approaches to achieve a Mandatory Access Control (MAC). However it is the security implications of the different virtualization strategies need to be a considered gives the combination of both options as a suggested scenario for improved security in the end.

The empirical literature comparing different containerization systems is limited, however, the following provide an outline some of the findings from recent comparing different platforms and some of the findings of these studies.

One of the ways that different Docker platforms can be considered is to compare the differences in the internal and external security of each platform. In Reshetova, Karhunen, Nyman, and Asokan (2014) , they use the system and attacker model to examine different factors that may impact security. These include issues such as process and filesystem isolation, and the limitation of resources. The results of this research highlights that although containers offer better performance, the are argued as  being less secure than VMs, but the default configuration of Docker is 'fairly secure'. Their suggestion for increased security is to combine the virtualization solutions.

In an additional study by Chung and Nah (2017) they provide a performance comparison between virtual clusters on Xen and Docker. Within this study they execute a programme called MapReduce within both environments. Within this study they execute everything on a single server running either several virtual machines on top of the hypervisor or containers on top of the Docker Engine. Different parameters like block sizes and virtual node numbers are adjusted to get richer insights. Based on this study they found that Docker cannot be considered faster than the other solution in each of their test cases, and they recommend using both options simultaneously.

In Shetty, Upadhaya, Rajarajeshwari, Shobha, and Chandra, (2017), they empirically compared to the performance of a bare metal server running the same applications. Within this study the overhead of the virtualization solutions Docker and OpenStack is compared. Based on the Phoronix test suite, benchmarks are executed and the CPU, memory and disk performance is compared. The study found that while the CPU and memory workloads are comparable amongst the virtualization solutions, the disk input / output performance is significantly better with Docker. In this study, they highlight the security related issues of Docker containerization (because of the shared kernel of the host system) in comparison to VMs which have a higher security level because of the distinct guest OS kernel and the hypervisor layer.

Based on this small review of the literature, the majority of research found generally only examine Linux based systems. This is probably due to the fact that Microsoft has only just adopted the Docker suite (just a year ago) and therefore there is limited literature concerning Windows and Docker (Docker Inc., 2016b).

## 3. METHODOLOGY

Design Science is chosen as methodology as 'In the design-science paradigm, knowledge and understanding of a problem domain and its solution are achieved in the building and application of the designed artifact' (Hevner, March, Park, & Ram, 2004).

From the review of the literature, it is apparent that when empirically comparing different solutions a quantitative approach is typically adopted. This approach leads to useful findings concerning the performance of systems but not necessarily according to the compatibility. In addition, to prove an empirical approach to test the system capabilities we also need to run the test in a specific configuration which can be comparable in setup.

The findings of this study are based on observations by the researcher while developing that artefact and focus on comparing, 1) the installation methods, 2) container types, 3) the cost of each solution, and 4) the approach to the virtualization and the currency of the software

The next section provides and overview of the test setup which will be used to compare the findings of this study.

### 3.1 Test Setup

The hypervisor is the basis of all further investigation as it runs the two VMs. Within the VMs, the operating systems of interest are installed and each of them runs Docker (namely a Linux and Window platform). Please see Figure 1 for a schematic of these two test environments.

The container is launched by the Docker deamon and all of those components mentioned combined are the artefact which is developed.

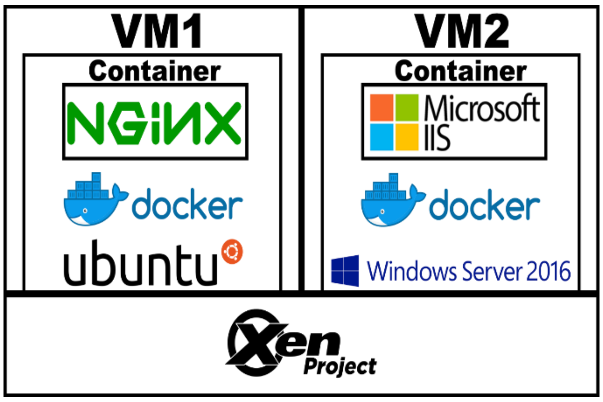

**Figure 1 Schematic test environment**

Testing was conducted in a managed environment to establish the closest balanced results. This was achieved using the following hardware and software parameters.

### 3.2 Hardware

A single server with 8 GB of RAM, 500 GB of storage and a Gigabit Ethernet connection was used. The CPU is a quad core Intel Xeon E5410 processor. Each of the VMs is equally set up with 1 CPU core, 2 GB of RAM, 50 GB of hard drive storage and a linked network adapter to the connection of the hypervisor.

### 3.3 Software (OSs)

Figure 1 shows the test environment with a Xen hypervisor underlining two VMs. VM1 is set up with Ubuntu Server 16.04 and VM2 is running Windows Server 2016 v1607, with Docker CE 17.09 running on both. Docker CE was chosen based on budgetary limitations. For testing purposes, a NGINX web server was the chosen application for VM1 and the Internet Information Service (IIS) for VM2.

## 4. FINDINGS

The following outlines the findings when comparing the two setup environments.

### 4.1 Comparison of Installation Methods

The most obvious difference between the two installation methodologies is the fact, that all installation commands on the Linux side are exclusively command line based, whereas the installation of Docker CE on Windows fully relies on the Graphical User Interface (GUI) and the pre-packaged installer.

The GUI based version needs less background knowledge and is straightforward. The installation file needs to be downloaded and executed and all following steps are guided, including modifications of system settings like activating the Hyper-V components. The command line-based installation on Linux requires a basic understanding of different components such as the Advanced Package Tool (apt), the related package lists and digital signing of software parts. The concept of command line-based tools, their parameters and how they are forwarded should be understood.

Apart from the manual installation on Linux, a scripted version is provided which automatically detects the system environment and the architecture it runs on and requests the necessary software components. That simplifies the installation process dramatically.

Businesses also have to take into account that the installation and maintenance of a Windows Server differs from a Linux server and their employees may have to build up additional knowledge before they are able to configure it properly for a production environment.

### 4.2 Container Types

The container OS image is the first layer in potentially many image layers that make up a container. This image provides the operating system environment for Docker (Microsoft, 2016). Depending on the OS used, two types are available: Windows based and Linux based container. While it is possible to run both container types on a Windows host, Linux hosts only supports Linux containers. During the investigation, it was determined that a Linux container on a Windows host can lead to unsupported states.

### 4.3 Cost comparison

Table 1 compares the costs of running Docker on Linux and Windows. The first option is to purchase the license for a Windows Server 2016 and get the Docker version free of charge. The other one is to go for the freely available Ubuntu Server but paying the yearly subscription fee for Docker then. In the proposed scenario, running Docker on Ubuntu would exceed the cost of the first one within four years.

| | Software Costs | |
|---|---|---|
| | **Windows Server 2016** | **Ubuntu Server 16.04 LTS** |
| **Operating System** | NZD 2.383,10 | NZD 0,00 |
| **Docker EE Basic (Yearly)** | NZD 0,00 | NZD 750,00 |
| | | |
| **SUM** | NZD 2.383,10 | NZD 750,00 |

Table 1: Software cost comparison

(Canonical Ltd., n.d.; Docker Inc., n.d.-a; Microsoft, n.d.)

The calculation is depended on the standard version of the Windows Server 2016 with a 5-Client Access License (Microsoft, n.d.). The pricing tables also shows higher prices for the Docker EE licenses for Linux servers as for their Windows counterparts throughout the whole range during the time the paper is written (Docker Inc., n.d.-a).

### 4.4 Nested virtualization

Because an additional layer of virtualization is present, Docker on Windows refused to start and an error message resulted. The default-running environment for Docker on Windows is Linux based, and this is referred to as *nested virtualization*.

The three main hypervisor solutions (VMware, Hyper-V and Xen) all offer options for nested virtualization (Poon & Mok, 2010; Qing, 2009; Thompson, Draga, & Cooley, 2016). Although a workaround can be found the official statement is that: 'Docker for Windows is *not* supported for nested virtualization scenarios' (Docker Inc., n.d.-c) - and this supports the errors experienced in this test. That leads to the problem that not all available virtualization solutions can be combined in every possible way.

### 4.5 Outdated Images

Several approaches to run a sample NGINX instance provided by Microsoft (Docker Inc., 2016b) within a Docker container on top of the Windows Server 2016 are not successful. First the 'latest' tag fails to pull as it is not used for this container image but is the standard one which is used if no other flag is specified via the parameters of the *pull* or *run* command. The tags section of the page shows three different types with a Windows Server core, a Windows Nano Server and a so called TP5 which has the equal size of the Nano Server tag and it is assumed that it points to the same image then. When the appropriate tag is applied to the pull command it is possible to download and start the container later on with the run command as well. But it immediately comes up with an error message due to tan outdated time stamp.

## 5. DISCUSSION

Based on these finding the following issues are highlighted.

### 5.1 Comparison of Installation Methods

The difference between running Docker on a Windows and a Linux host OS need to be considered. First, the installation methods differ from a command line based approach on Linux to an exclusively GUI on Windows. The Windows GUI is straightforward, and all steps are guided. The commands on Linux seem more difficult to deal with as every single command can fail which has to be investigated individually, but as a scripted version is available which also checks the system environment automatically and adjusts the necessary parameters, it can be considered as straightforward as well. Docker itself is also command line driven, therefore it is assumed that the target audience has the required skill set to handle the installation process as well. Overall the installation on both systems is trivial. The factor which has to be taken into account while deciding is more the configuration and maintenance of the host system itself.

### 5.2 Container Types

The compatibility of the two different available types of containers leads to another decision while choosing the right setup. While Windows should be able to execute both types, it can lead to difficulties in a specific nested virtualization scenario as described in section 4.4. Linux is just able to run Linux containers, but it works really reliable.

### 5.3 Costs

Costs are of course also a factor where the two host operating systems differ. At the time of writing, Docker offers a free EE license to Windows Server 2016 customers, whereas Linux customers are charged a yearly fee. This leads to the amortization of the initial licensing costs of the Windows Server after 4 years compared to the Linux server, which is freely available.

### 5.4 Issues

Nested Virtualization is the biggest issue faced during this research and can affect everyone that tries to combine virtualization approaches in such a way. This should be taken into account when deciding on a software and hardware stack for running Docker. If existing virtualization solutions are used, this is one of the main points to consider.

The availability and maintenance of existing images as described in section 4.5 seems to be better on the Linux side. This aspect is important, because one of the benefits Docker adds to the containerization approach is that existing solutions can be shipped, shared and reused easily. If a business can build on existing Docker images of great diversity it is more efficient than starting from scratch.

## 6. IMPLICATIONS OF RESEARCH

Containerization in a Linux OS has an advantage to educational institutions and businesses because; (1) it is a proven global adoption historically and; (2) it is Open Source structure which gives developers, administrators and programmers access to core code.

In comparison, Microsoft is a newcomer to introducing the technology and this research demonstrates that although the immediate installation methods via GUI is straightforward, overcoming obstacles is restricted and needs to be considered. Since the research was conducted, Microsoft has released Windows Server 2016 v 1709 which removes the GUI to support remote management, is the first of the new semi-annual release strategies, and is designed for cloud deployment (Microsoft, 2017). According to Docker, Windows Server 2016 v1709 does not support some of the core container components on all platforms (Docker, 2017). This could possibly limit the expectations on businesses and result in the need for further research before implementations can be considered.

Overall the following key aspects are apparent when comparing these two platforms:

- The installation approaches for Docker on Linux and Windows differ, but both are straight forward. The maintenance of the host system itself needs to be considered as well.
- Different container types exist for Linux and Windows and some combinations lead to a nested virtualization scenario, which is not officially supported by Docker and should not be used in a production environment.
- The initial costs of the Windows host solution are exceeded by yearly costs for the Linux host after 4 years.
- The issues faced are mainly the nested virtualization as described above. Apart from it outdated and poorly maintained images are found as well.

## 7. CONCLUSIONS

This research provided a comparison between two platforms (Linux and Windows) using Docker for containerization. It provided an initial understanding on the differences between these two platforms. This study will help to provide some initial insights into an area with limited empirical research. .

The scope of this study was however, limited to a self-hosted server scenario and all kind of cloud services offered online are not taken into account, although the results may be transferable to cloud services. Free software licenses were used in favour of paid or subscription based models.

Docker is a relatively new technology and also in active development while this paper was written. All findings must, therefore, be interpreted with the available capabilities of the technology during the investigation period.

Future research within the area of Windows, as a host system, is needed. With limited literature and experimental documentation around that, Linux and Docker have been the main container solution until now. Making this a part of empirical research we ensure to provide an unbiased view to the issues related to these two platforms and opens up future potential research as these two platforms become more popular.

## 8. REFERENCES

Bui, T. (2015). Analysis of docker security. *arXiv preprint arXiv:1501.02967.*

Canonical Ltd. (n.d.). Plans and pricing. Retrieved 2017-10-23, from https://www.ubuntu.com/support/plans-and-pricing

Chung, H., & Nah, Y. (2017). *Performance Comparison of Distributed Processing of Large Volume of Data on Top of Xen and Docker-Based Virtual Clusters.* Paper presented at the International Conference on Database Systems for Advanced Applications.

Datadog Inc. (2017). 8 Surprising Facts About Real Docker Adoption. Retrieved 2017-10-18, from https://www.datadoghq.com/docker-adoption/

Docker Inc. (2017), What is a Container? Retrieved March 27, 2018, from https://www.docker.com/what-container

Docker Inc. (2016a). Docker Announces Commercial Partnership with Microsoft to Drive Adoption of Containerized Applications in the Enterprise. Retrieved 2017-10-26, from https://www.docker.com/docker-news-and-press/docker-announces-commercial-partnership-microsoft-drive-adoption-containerized

Docker Inc. (2016b). Public Repository microsoft/sample-nginx. Retrieved 2017-10-23, from https://hub.docker.com/r/microsoft/sample-nginx/

Docker Inc. (Producer). (2017, 2017-10-21). Docker Online Meetup: Announcing Docker CE + Es. (2014). *End Note [Computer Software]*. Retrieved from End Note: http://endnote.com/

Docker Inc. (2017) Preview Docker for Windows Server 1709 and Windows 10 Fall Creators Update. Retrieved 2018-03-12 from https://docs.docker.com/install/windows/ee-preview/

Docker Inc. (n.d.-a). Docker Pricing. Retrieved 2017-10-21, from https://www.docker.com/pricing

Microsoft Inc. (2017) Introducing Windows Server, version 1709. Retrieved 2018-03-12 from https://docs.microsoft.com/en-us/windows-server/get-started/get-started-with-1709

Microsoft. (n.d.). Buy Windows Server 2016 Standard - Microsoft Store New Zealand. Retrieved 2017-10-23, from https://www.microsoft.com/en-nz/store/d/windows-server-2016-standard/dg7gmgf0ds12/0004

Cheremisina, E. N., Belov, M. A., Tokareva, N. A., Nabiullin, A. K., Grishko, S. I., & Sorokin, A. V. (September, 2017). Embedding of containerization technology in the core of the virtual computing lab, 4. Proceedings of the XXVI International Symposium on Nuclear Electronics & Computing (NEC'2017) Becici, Budva, Montenegro.

Wager, E., & Kleinert, S. (2011). Responsible research publication: International standards for authors. In T. Mayer, & N. Steneck (Eds.), *Promoting research integrity in a global environment* (pp. 309-316). Singapore: Imperial College Press / World Scientific Publishing.