Overall comparison of the performance loss of virtualization

Research Project for Large Systems
Master Security and Network Engineering
University of Amsterdam

Project Report
Version: 1.0

Alexander Blaauwgeers
alexander.blaauwgeers@os3.nl

Kotaiba Alachkar
kotaiba.alachkar@os3.nl

December 30, 2017

Abstract

This project has been conducted to evaluate performance loss of virtualization on virtual servers compared to physical server. In this project we will focus on medium size e-commerce websites. Our main target for performance testing is the website response time.

During the duration of the project a number of experiments were conducted. Started from a complex environment with a distributed web application and later brought back to a simple static web page. These experiments simulated a real life users experience scenario.

After going going through the project we found that the loss of performance depends on the application architecture. It differs 5 milliseconds with a simple application up to 2 seconds for a complex application.

Contents

1 Introduction

In the past people thought of "servers" as big, bulky machines installed in data centers. Now, "servers" more often mean cloud instances or VPSs. Although physical servers are still a favorite choice for many. However, research shows that dedicated server market grows by $237 million each year [5].

There are several factors that determines which infrastructure is suitable for a given business but in this research we will focus only on the performance factor and specifically the performance loss of virtualization. The method used in our research is to implement a real environment setup using a virtual servers and an identical setup using four physical servers. Then, conduct an overall comparison between these two setups. We expect after this research is to have a clear view and a comprehensive understanding about virtualization in term of performance and identify how performance is affected by virtualization.

1.1 Research questions

Our main research question is as follows:

How do virtual servers compared to physical servers differ in term of performance?

This main research question is supported by two sub-research questions:

- *Will be there any performance loss of virtualization?*
- *Which is more suitable for a mid-size business infrastructure?*

1.2 Related work

There are two existing work related to the work presented in this research. The former work done by a group of researchers from Karlsruhe Institute of Technology, Germany. They conducted an experimental state-of-the-art virtualization platforms, Citrix XenServer 5.5 and VMware ESX 4.0, as representatives of the two major hypervisor architectures. Based on the results, they propose a basic, generic performance prediction model for the two different types of hypervisor architectures. Their target was to predict the performance overhead for executing services on virtualized platforms [2].

The latter work was about the impact of virtualization on network performance of amazon EC2 data center done by a group of reserarcher from Rice University, USA. They presented an empirical measurement study on the end-to-end networking performance of the commercial Amazon EC2 cloud service, which represents a typical large scale data center with machine virtualization. The focus of their study is to characterize the networking performance of virtual machine instances and understand the impact of virtualization on the network performance experienced by users [6].

1.3 Research Scope

This research will focus on performance loss of virtualization on virtual servers compared to physical servers. Furthermore, the starting point of our research is to keep it practical and simulate real environment infrastructure on both implementation and traffic wise. If possible we might expand our research to also include solutions to avoid any possible performance loss of virtualization. In addition to cost, time, and administrative management performance comparison between these two setups.

2 Experiment

This experiment followed strict patterns to ensure that it would not affect the servers of each of the tested solutions. No real data was used in the tests. Each solution got an account created and the same set of dummy data inserted into it.

2.1 Testing environment

The research will be conducted within a client-server environment. In this research we will be using five servers with the following specifications:

- Intel Xeon processor
- 8GB ram
- SSD + HDD for storage

2.1.1 Physical server

We used four physical servers to setup the needed services. Additional tools will be used, as well multiple logging and bench marking tools that allow us to keep a detailed log on the resource consumption and performance measurement. The fifth physical server is used for building the virtual setup.

2.1.2 Virtual server

For the virtual server we will be using Xen 4.8.2 as hypervisor to implements the VMs. On top of the hypervisor there is a Dom0.

2.1.3 Setup overview

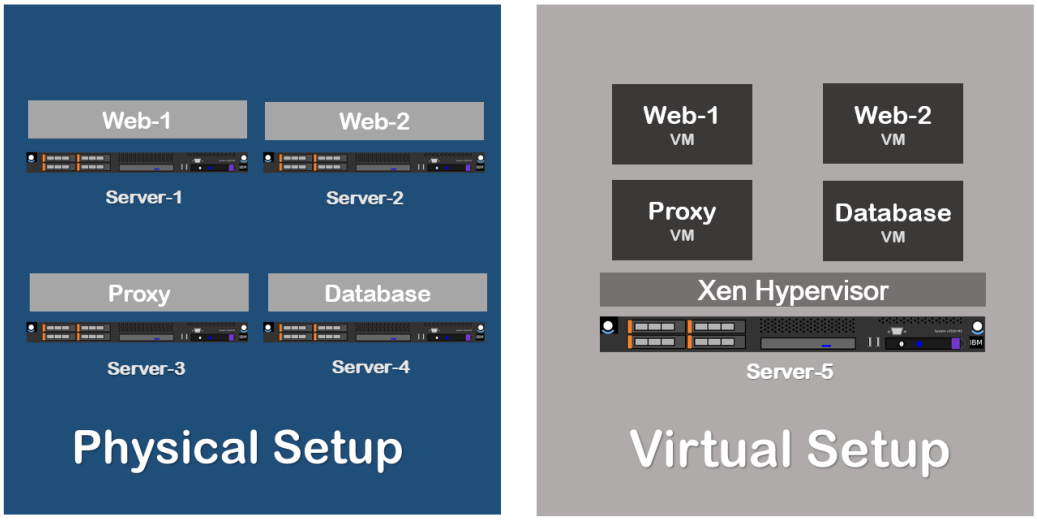

Figure 1: Test Environment Setup

The **figure 1** above illustrates both the physical and virtual setups environment, in addition to each service that run on which server. The networking between the servers on the physical setup is also physical. The networking between the virutal servers is virtualized by using a Xenbridge.

2.2 Web application

For the application part of our setup, we used osCommerce[3], as shown in **figure 2**. OsCommerce is an opensouce webstore conternt management used in many webshops. It runs on a webserver and uses PHP and MySQL. As webserver we used Apache (Apache/2.4.18) with the PHP (7.0.22-0) extention. For the database we used MySQL(5.7.20-0). The proxyserver runs nginx(1.10.3). The setup for the Hardware and VM are the same. It provides a realistic web application.

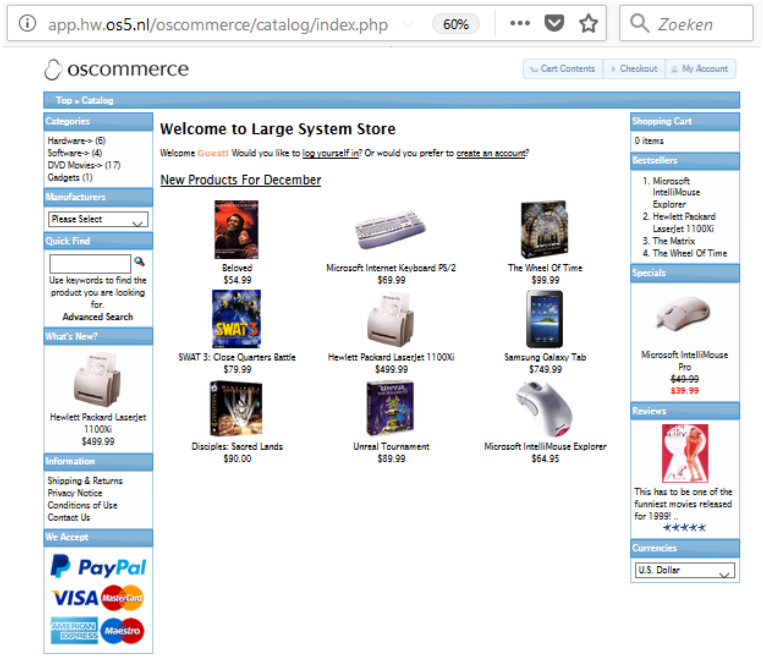

Figure 2: Large System Store Powered by osCommerce

2.3 Tools

In order to achieve a consistent and precise output for our experiments. We used two different tools run on different operating systems to load and stress our setup. The tools are the following:

- The former one is **Apache HTTP server benchmarking tool** which is an open source Linux based tool. It is "a tool for benchmarking your Apache Hypertext Transfer Protocol (HTTP) server. It is designed to give you an impression of how your current Apache installation performs. This especially shows you how many requests per second your Apache installation is capable of serving" [1].

- The latter one is **Paessler Webserver Stress Tool** which is Windows based free performance, load, and stress-test for web servers. It is a powerful HTTP-client/server test application designed to pinpoint critical performance issues in your web site or web server that may prevent optimal experience for your site's visitors. It simulates "the HTTP requests generated by hundreds or even thousands of simultaneous users, you can test your web server performance under normal and excessive loads to ensure that critical information and services are available at speeds your end-users expect" [4].

2.4 Methodology

In this project we followed a methodology that starts from basic tests to increasingly complex ones on both virtual and physical servers. The steps we followed were:

- **Basic setup tests:** In the basic setup tests we used only the Apache HTTP server benchmarking tool to experiment the web server alone by itself without the complete setup as illustrated in the **figure 3**. We targeted only the static web page of our osCommerce store.

Figure 3: Basic Setup Test Methodology

- **Full setup tests:** For the complete setup test we used both the Apache HTTP server benchmarking tool and Paessler webserver Stress tool to experiment the complete setup environment which includes proxy, web, and database servers as shown in **figure 4**. We targeted the proxy server which acts as load balancer between our two webservers. And these two webservers interact directly with the database server. Which leads to a complete setup tests.

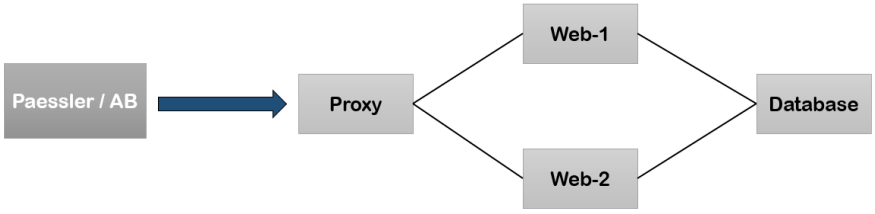

Figure 4: Full Setup Test Methodology

2.5 Naming structure

For the project we used the following url structure, to prevent mistakes during the measurement. Fully Qualified Domain Name (FQDN) of the servers used:

$$\{app \mid web1 \mid web2 \mid db\} . \{vm \mid hw\} .os5.nl.$$

For the static web page:

$$\begin{bmatrix} http://\{app \mid web1 \mid web2\} . \{vm \mid hw\} .os5.nl \\ \vdots \\ http://web1.hw.os5.nl \end{bmatrix}$$

For the dynamic web page:

$$\begin{bmatrix} http://\{app \mid web1 \mid web2\} . \{vm \mid hw\} .os5.nl/oscommerce/catalog/\{phpscript\} \\ \vdots \\ http://app.vm.os5.nl/oscommerce/catalog/product\_info.php?products\_id=25 \end{bmatrix}$$

3 Results

In this section the results of the experiments preformed as part of the project. The results will be presented per application and, if required, are divided into several test runs if required. The tests have been performed several times. The experiment have been performed on both the virtual and hardware environment.

3.1 Paessler Webserver Stress Tool

The experiment has been preformed multiple times, with the same scenario. Afer each experiment the database referted the original state. Each user has made multiple requests to a balanced set of application functions, based on the shopping scenario, triggering different kind of transactions. Requests got timed-out after about 1500ms. A scenario to simulate realistic transactions on the full-setup webshop. Has been preformed using the steps shown in **figure 5**, **figure 6** and **figure 7** shown below. An example of such scenario script can be found in Appendix 3 (listening 6).

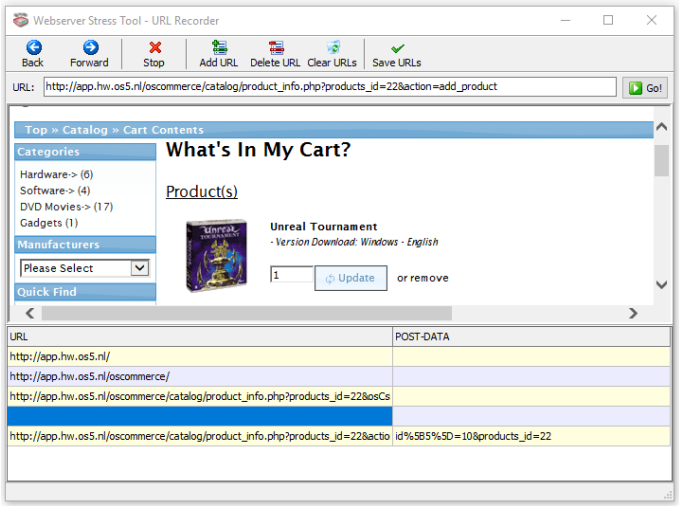

Figure 5: URL recording

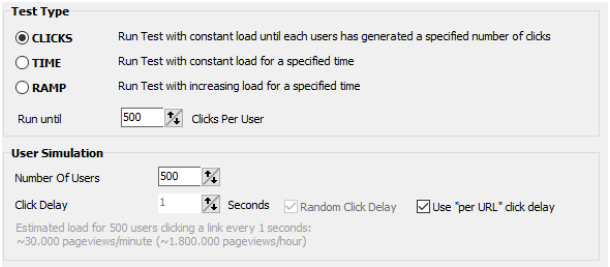

Figure 6: Test characteristics and userdelay

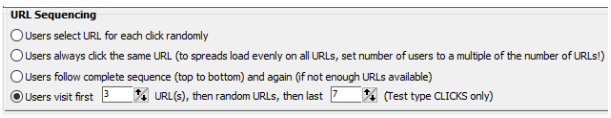

Figure 7: URL sequencing

3.1.1 Result: Measuring requests & transferred data

First the result of the request and transferred data are shown below. The hardware in **figure 8** and for the VM we refer to **figure 9** which show the number of open requests as well as the number of sent and received requests in comparison with the network traffic on both the virtual and physical setups.

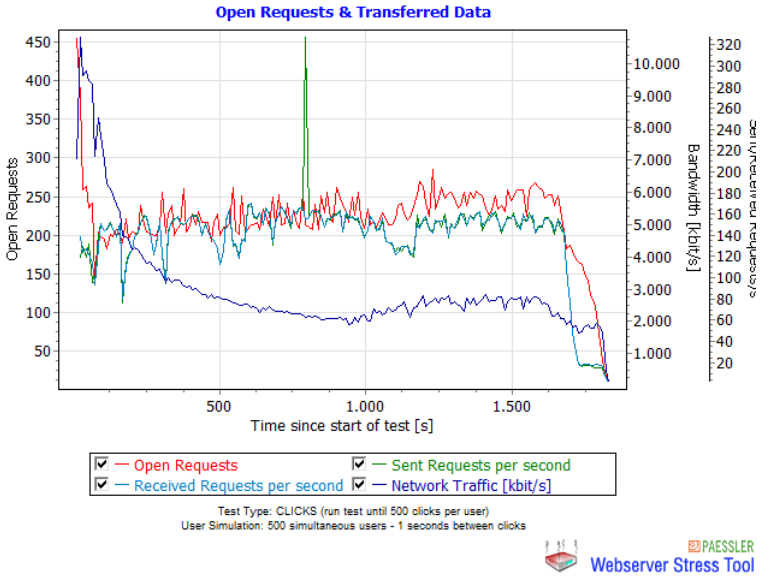

Figure 8: Requests & Transferred Data for Physical Setup

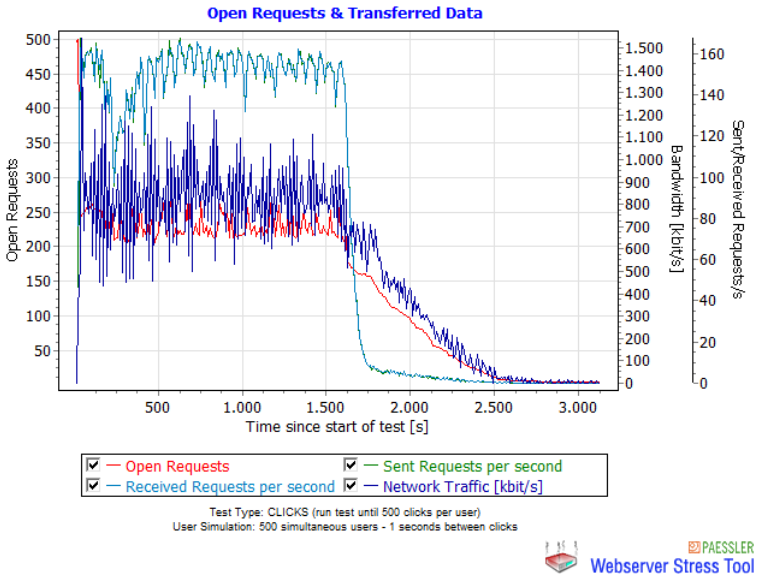

Figure 9: Requests & Transferred Data for Virtual Setup

The 500 users we nicely balanced in the hardware setup, while on the virtual setup they were handled all at the same time. A large amount of requests on the virtual setup timed-out before they were handled by web server.

3.1.2 Result: Measuring transferred Data & system Memory & CPU load

The second result is about the transferred data. Below in **figure 10** for the hardware and for the VM we refer to **figure 11**, which describe the measurement for vital parameters of the machine it runs on. It can be helpful to find out if the limits of the test client have been reached testing both the virtual and physical setups:

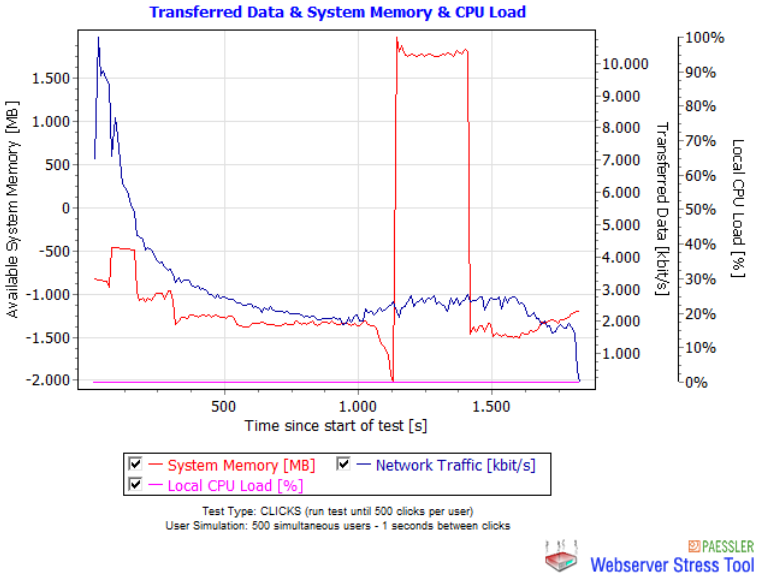

Figure 10: Transferred Data & System Memory & CPU Load for Physical Setup

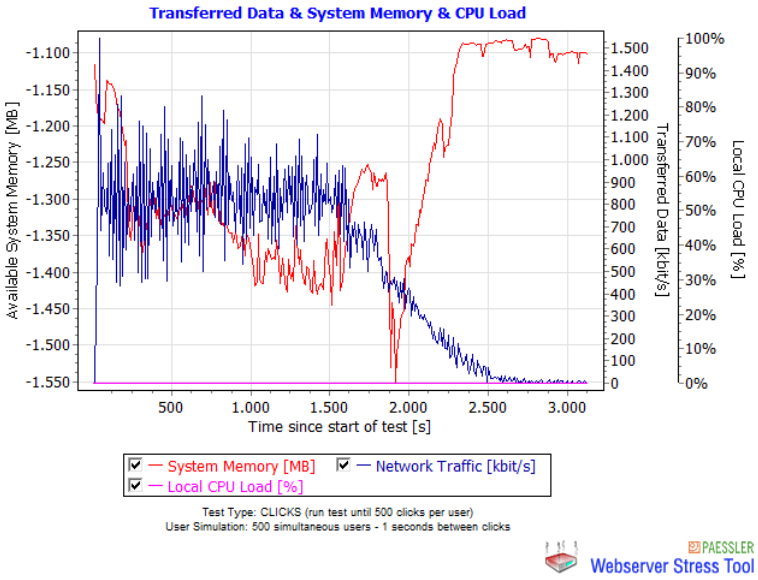

Figure 11: Transferred Data & System Memory & CPU Load for Virtual Setup

The 500 users on the hardware setup are handled smoothly in order. There is a continuous rate of requests handled by the server. During the experiment on the virtual setup there was much more requests queued in memory. The response were handled two seconds later by the client.

3.1.3 Result: Measuring Click Time, Hits/s & Clicks/s

Figure 12 & **figure 13** illustrate the average time a user waited for his request to be processed (including redirects, images, frames, etc., if enabled), the hits per second, and the users per clicks.

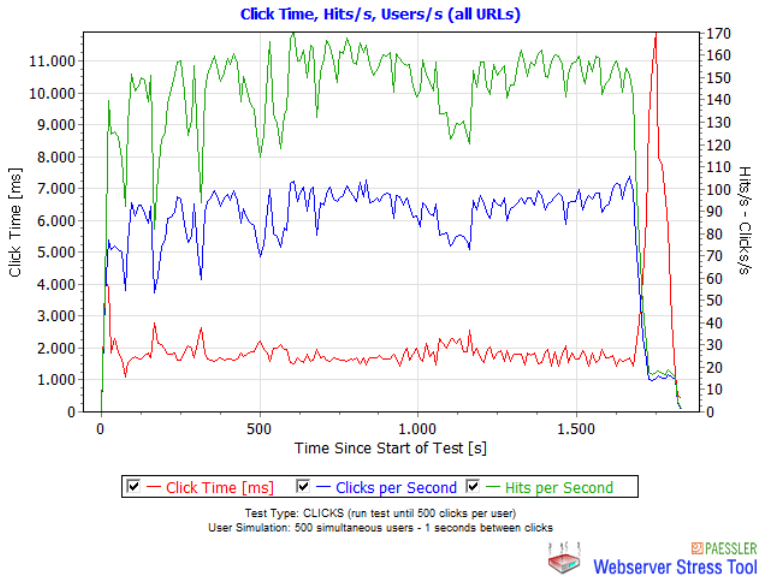

Figure 12: Click Time, Hits/s & Clicks/s for Physical Setup

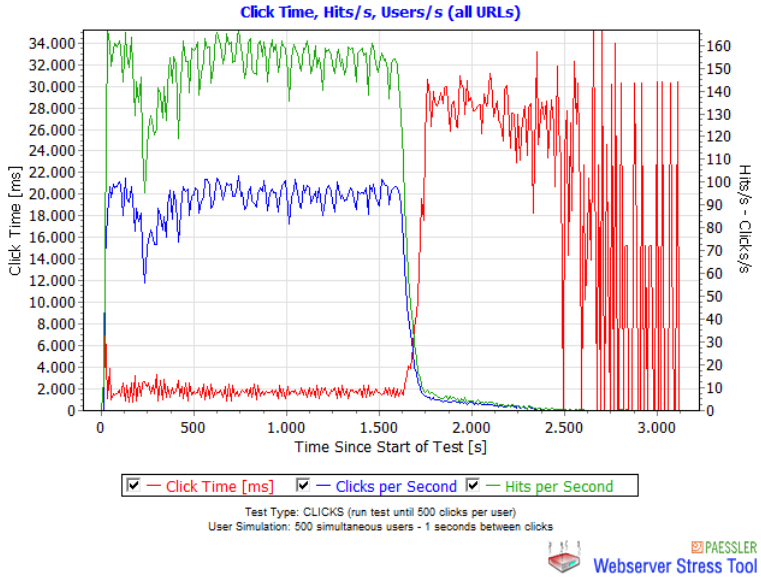

Figure 13: Click Time, Hits/s & Clicks/s for Virtual Setup

We can see that with 500 users the two lines for “clicks per second” (blue) and “hits per second” (green) differ more and more. The reason is that hits includes requests that produce errors, but clicks are only calculated from the requests that were successful.

3.2 Apache Benchmark

- Test 1: Full-Setup
  - A load of similar transactions on the full-setup webshop.
  - Due to the nature of the webshop, the responses are slightly used specific. Because functions like the shoppingcart-session.
  - Generates multiple read and write transactions to the database.
  - Makes use of a round-robin load balancing reverce proxy based on ngnix.
- Test 2: Webserver-only
  - A load of similar transactions on apache webserver only.
  - Generates multiple requests for a specific resource.
  - Does not make use of the proxy and/or the database.
- Test 3: Optimized CPU and Memory
  - A combination of test 1 and 2 with a optimized CPU and memory mix.

3.2.1 Test 1: Full-Setup

Test with AB to Proxy displayed in **figure 15** & **figure 15**. As described in the bullets above *(3.2)*. The first experiment uses 4 servers and the target is the catalog of the webshop at the proxy server.

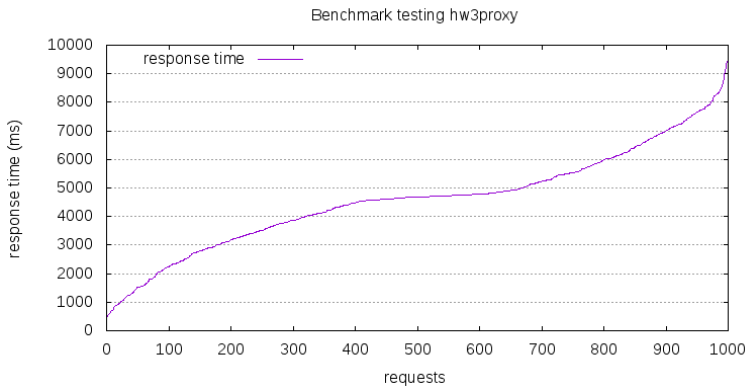

Figure 14: Test 1: Full-Setup for Physical Setup

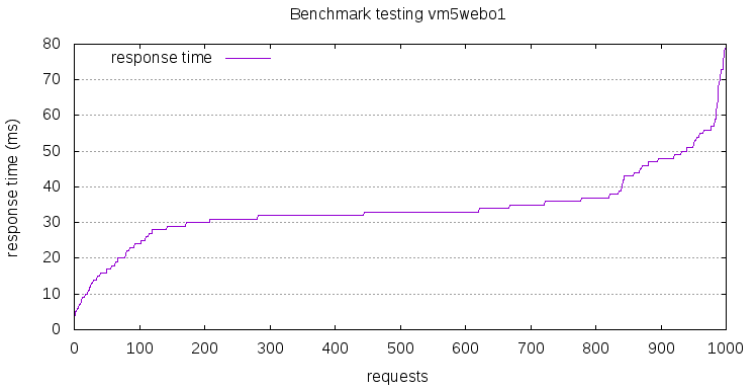

Figure 15: Test 1: Full-Setup for Virtual Setup

3.2.2 Test 2: Webserver-only

Test with AB to Webserver displayed in **figure 17** & **figure 17**. As described in the bullets above *(3.2)*. The second test has been preformed to test the server without the application, database transactions and calculations.

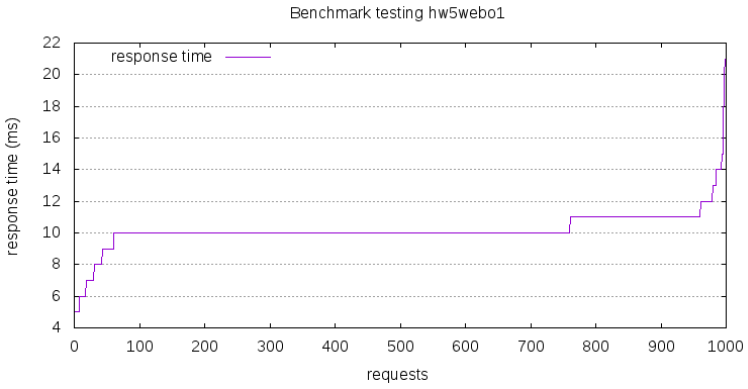

Figure 16: Test 2: Webserver-only setup for Physical Setup

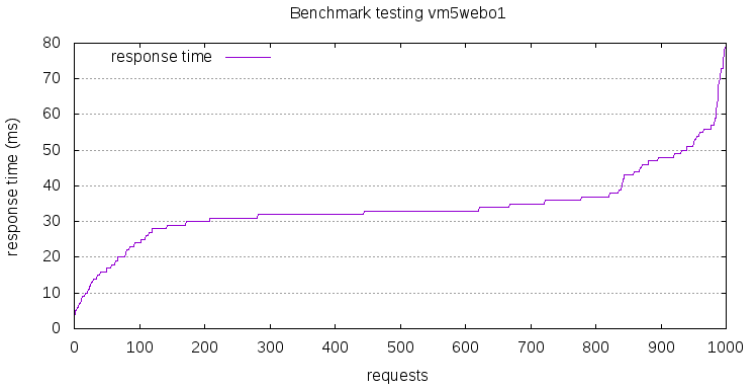

Figure 17: Test 2: Webserver-only setup for Virtual Setup

3.2.3 Test 3: Optimized CPU and Memory

Test with optimized CPU and Memory, with both setup, on the VM. Seen in **figure 19** & **figure 17**. As described in the bullets above *(3.2)*. The last test has been preformed to see the inpact of the configuration.

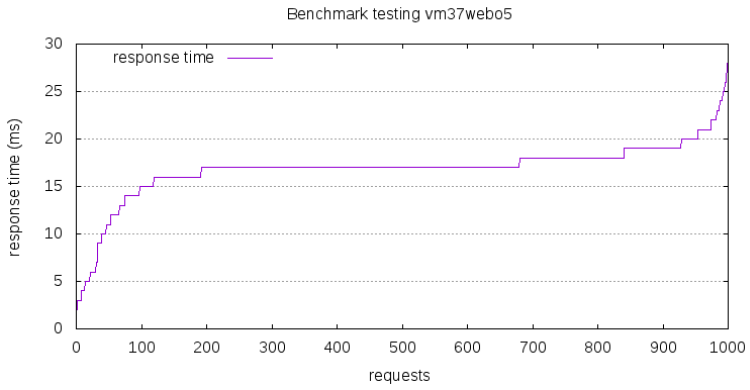

Figure 18: Test 3: Optimized CPU and Memory setup for Full VM Setup

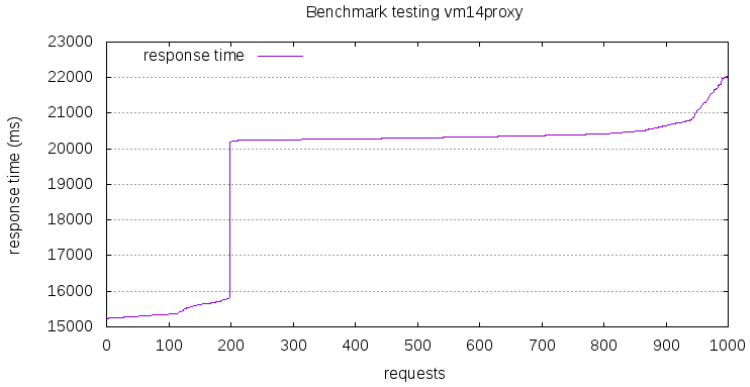

Figure 19: Test 3: Optimized CPU and Memory setup for Web-only VM Setup

4 Discussion

During the experiments we saw large differences between the physical and virtual setup. The largest difference is seen on the full-setup.

The hardware setup is stable and is functioning quite normal with a responce time of the catalog of 5000ms, but the virtual setup is much more slower with a response of 20.000ms. That is 3 times slower and unacceptable. This might be the effect of distribution transparency.

After the experiment with the full application setup we also preformed a test with a webserver only. In this experment with more simple and limited setup we noticed a difference of 5ms between the hardware and VM. In **figure 17**, we see the hardware has an average of 10ms and in **figure 19** the pure VM has an average delay of 15ms under the same circumstances with the same experiment. We consider that the pure loss of virtualization.

In the last test we tried to minimize the difference by optimizing and doubling the CPU and memory assigned to the VM. There was almost no difference visible in performance.

We notice that the application performance, in a virtual environment, is strongly affected by the architecture of the application.

The webshop has a stateless design where the session date (e.g. shopping-cart) are stored in the datebase. Session related to an session_id saved in a cookie at the client-side which reffers to a record in the database.

5 Conclusion

There is a performance loss of virtualization based. The total loss of performance is highly influenced on the setup and architecture of the applications which are running. When an application is not optimized for virtualization the performance may drop significant. In a more simple static website setup we notice performance difference of 5ms.

We consider that 5 milisecond the pure loss of virtualization.

5.1 Future Work

As future work for our project we might enrich and improve the contents and the output of this research by applying few ideas, such as using othe applications such as **Vmware** or **Virtual Box** or any other proprietary software solutions for virtulization. In addition to experiment the setup on different operating system such as **Windows**. We might also use more variance of hardware to build the setup environment. Another important idea is to make more intense and larger test setup with longer measurement time. These ideas help with furthering the findings on these password managers and could open the doors to new findings.

References

[1] *ab - Apache HTTP server benchmarking tool.* URL: https://httpd.apache.org/docs/2.4/programs/ab.html.

[2] *EVALUATING AND MODELING VIRTUALIZATION PERFORMANCE OVERHEAD FOR CLOUD ENVIRONMENTS.* URL: http://www.academia.edu/download/39830393/Evaluating_and_Modeling_Virtualization_P20151109-29150-1kc1ubg.pdf.

[3] *osCommerce.* URL: https://www.oscommerce.com.

[4] *Paessler - Webserver Stress Tool.* URL: https://www.paessler.com/tools/webstress.

[5] *Size of the dedicated hosting market worldwide from 2011 to 2019 (in billion U.S. dollars).* URL: https://www.statista.com/statistics/500551/worldwide-dedicated-hosting-market-size/.

[6] *The Impact of Virtualization on Network Performance of Amazon EC2 Data Center.* URL: http://ieeexplore.ieee.org/abstract/document/5461931/?part=1.

[7] *Virtualization Ask The Experts 1.* URL: https://www.anandtech.com/show/3827/virtualization-ask-the-experts-1.

6 Appendices

```
#!/bin/bash
clear
echo "Start of experiment No. 4"
for i in {1..25}
 do
 echo "Round $i"
 echo "=== Round $i ===" >> abresult_hw.txt
 ab -n 1000 -c 100 -g abresult_hw_$i.tsv "http://app.hw.os5.nl/oscommerce/
    catalog/" >> abresult_hw.txt
 sleep 1
 done
vim abresult_hw.txt -c "hardcopy > abresult_hw.pdf | q"
echo "Result has been saved."
```

Listing 1: Appendix 1 - template bash code for ab

```
set terminal png size 750,400
set size 1, 1
set output "abresult_hw_1.png"
set title "Benchmark testing hw1webo4"
set key left top
set grid y
set xlabel 'requests'
set ylabel "response time (ms)"
set datafile separator '\t'
plot "abresult_hw_1.tsv" every ::2 using 5 title 'response time' with lines

set terminal png size 750,400
set size 1, 1
set output "abresult_hw_2.png"
set title "Benchmark testing hw2webo4"
set key left top
set grid y
set xlabel 'requests'
set ylabel "response time (ms)"
set datafile separator '\t'
plot "abresult_hw_2.tsv" every ::2 using 5 title 'response time' with lines
......
```

Listing 2: Appendix 2 - template script for gnuplot the result

```
0,0,URL#
1,0,Name
2,0,Click Delay [s]
3,0,URL
4,0,POST data (or @filename@)
5,0,Username $i[username]
6,0,Password $i[password]
3,1,http://app.vm.os5.nl/oscommerce/catalog/
0,2,2
3,2,http://app.vm.os5.nl/oscommerce/catalog/product_info.php&products_id=$i[
    item_id][0]
0,3,3
3,5,http://app.vm.os5.nl/oscommerce/catalog/product_info.php?action=
    add_product&products_id=$i[item_id][0]
0,6,6
3,8,http://app.vm.os5.nl/oscommerce/catalog/product_info.php?cPath=2_19&
    products_id=$i[item_id][0]
0,9,9
3,11,http://app.vm.os5.nl/oscommerce/catalog/index.php?cPath=2_20
0,12,12
3,12,http://app.vm.os5.nl/oscommerce/catalog/product_info.php?cPath=2_20&
    products_id=$i[item_id][0]
3,16,http://app.vm.os5.nl/oscommerce/catalog/shopping_cart.php?action=
    update_product
4,16,id%5B26%7B3%7D8%5D%5B3%5D=8&cart_quantity%5B%5D=5&cart_quantity%5B%5D=
    $i[item_aantal][0]&products_id=$i[item_id][0]
3,18,http://app.vm.os5.nl/oscommerce/catalog/product_info.php&products_id=$i
    [item_id][1]
3,21,http://app.vm.os5.nl/oscommerce/catalog/product_info.php&products_id=$i
    [item_id][1]&action=add_product
4,21,products_id=4
3,22,http://app.vm.os5.nl/oscommerce/catalog/shopping_cart.php?products_id=
    $i[item_id][1]&action=remove_product
0,23,23
3,23,http://app.vm.os5.nl/oscommerce/catalog/checkout_shipping.php
0,24,24
3,24,http://app.vm.os5.nl/oscommerce/catalog/login.php?action=process
4,24,formid=$i[card_id]&email_address=testuser@os5.nl&password=test123
0,25,25
3,25,http://app.vm.os5.nl/oscommerce/catalog/checkout_shipping.php
4,25,formid=$i[card_id]&action=process&shipping=flat_flat&comments=I+do+not+
    really+want+this+%2C+only+test
0,26,26
3,26,http://app.vm.os5.nl/oscommerce/catalog/checkout_confirmation.php
4,26,formid=$i[card_id]&payment=cod&comments=I+do+not+really+want+this+%2C+
    only+test
0,27,27
3,27,http://app.vm.os5.nl/oscommerce/catalog/checkout_process.php
4,27,comments=I+do+not+really+want+this+%2C+only+test
0,28,28
3,28,http://app.vm.os5.nl/oscommerce/catalog/logoff.php
```

Listing 3: Appendix 3 - Shopping Scenario