Project Terms of Reference — To compare Virtualisation and Containerisation for the context of hosting headless servers.

James Poxon

7th December 2020

1 Background

Virtualisation has been utilised by a number of industries for a long time now, with first iterations of virtual machines dating back to IBM in the 1960s [Pugh, 1995]. System Virtual machines allow an operating system to emulate the function of a full operating system layered on top of a base operating system. Functionally, this allows multiple different logical 'computers' with varying operating systems to run on one physical computer. In most modern implementations, virtualisation requires software (known as a hypervisor) to manage and create the virtual machines.

Whilst I was on a year-long placement provided as part of a sandwich course at my university, I had the chance to work in an IT risk department at a reputable enterprise in Newcastle Upon Tyne. whilst working there I witnessed first hand, infrastructure and operations departments using virtualisation for much of the internally and externally facing server infrastructure, in what was an unwieldy and cumbrous use of scripts to update and install patches and dependencies across a multitude of systems. This resulted in a number of incidents where systems had to be pulled down in order to update the Operating Systems of individual virtual machines. These methods often caused unnecessary down time for systems, resulting in substantial risk for the business. Furthermore, this method often put a substantial stress on the hardware of these systems, with hardware boxes often running at full resource potential under heavy load, and whilst these boxes were designed to withstand these kinds of loads, it still affected the longevity of the hardware.

In recent years there has been research into the use of containers instead of virtual machines for various operations across computing industries [Watada et al., 2019]. Containers are different from virtual machines in that they run on the base OS, and don't require a secondary emulated operating system to be managed by a hypervisor. This is a benefit as it reduces the resources [Joy, 2015] used by each instance. Overall, containers are a more lightweight and efficient system, that are easier to keep up to date. Whilst research has claim in displaying the benefits of containerisation, much of the server infrastructure of enterprise remains reliant on Virtualisation, not Containerisation.

I have also had the opportunity to partake in gatherings held by CoTech, a network of tech co-operatives that meet semi-regularly to discuss various tech related topics. Whilst I was there, it was discussed that a large portion of these small enterprises used Docker, a system for packaging and deploying containers, to develop applications for their clients.

There is historical research that compares virtualisation and containerisation for other technical applications, for example [Dua et al., 2014] compares the two methods inside the context of PaaS (Platform as a service), which is similar to co-techs use of Docker. It seems that the application for containerisation has been realised as early as 2014 when it is being applied to the development and hosting of applications, but not as much when talking about the running of wider server infrastructure. This is further supported by research from '451 Research' who in 2017 estimated a compound annual growth rate of 40% for containers in 2020 [Research, 2017], suggesting a coming paradigm shift from virtualisation to containerisation.

2 Proposed Work

I plan to do similar work to the studies I have already mentioned [Joy, 2015], [Dua et al., 2014], but instead focus this work on my own context and my own interest in Operating Systems and Server Infrastructure. I have experience with running headless servers on virtual machines already, through the Advanced Operating Systems module I did in my second year of study at Northumbria University.

It is important to set a baseline system here, and to best reflect a realistic topology, I will use LAMP (Linux + Apache + MySQL + PHP). I have experience with Ubuntu server, so I will use this version of Linux as the base operating system for my virtual machines, and plan to host a full working topology of servers, including a DNS architecture, a web-server architecture that includes HTTP (Apache) servers, NFS servers and MySQL servers. The reasoning for doing this is to create as realistic a topology as possible, and to ensure a somewhat realistic level of network traffic so that meaningful data can be captured.

For the virtual machines, I will use the virtual machine infrastructure available to me through the university, and for the container infrastructure, I plan on using Docker. This is because Docker is a popular choice among app developers, and is supposed to make container deployment easy. Though, as I have never worked with containers before, if Docker fails to be a good way of providing containers for web-servers, then there are a number of other ways to deploy containers, such as LXC, that I could turn to as a contingency.

I will also need to set a standard for measurement to ensure that my data is meaningful and uses the scientific method. I plant to use tools such as iPerf3 (for network performance) and Sysbench (for hardware performance) across a number of servers. It is important to measure both network and hardware performance (CPU, Memory, Disk, etc) to ensure that described benefits are achieved for the system as a whole. iPerf is an industry standard tool for measuring network performance on Linux systems, whilst Sysbench is a full benchmarking suite for linux that will let me benchmark the CPU, file IO, and importantly, MySQL performance (allowing direct measurement of database performance on the machine that will be hosting the previously mentioned MySQL database. Whilst I have mentioned these

benchmarking tools, it is possible that they could have problems with integrating with certain applications or environments, in this case, there is a number of other standard benchmark utilities (Phoronix Test Suite, KDiskMark, UnixBench, etc) available that should give me the flexibility to create measurements. Whilst these utilities all have differing overheads within them (meaning they will use up varying resources to run the benchmark), as long as I use the same benchmark across the same machines I am comparing, this overhead shouldn't effect the measurable performance difference between these machines.

Once data is collected, I will do a comparative analysis of the data to determine if there is a clear improvement as previous research suggests, and if this difference is enough to justify a change in the current best-practice use of virtualisation.

3 Aims and Objectives

3.1 Aims

> To compare and contrast the performance difference and hardware impact between virtualisation and containerisation when running headless servers.

3.2 Objectives

Your objective list is a series of measurable objectives, can you tick each one off as *done*? I usually expect between 8 and 12 objectives

1. **Explain the problem domain that encompasses current practices in virtualisation.**
2. **Explanation of the two different methods in practice.**
3. **Determine the software and hardware to be used.**
4. **Design the network infrastructure to be built.**
5. **Learn how to implement and utilise containers using Docker.**
6. **Build the network topology on the virtual machines.**
7. **Build the network topology on the container system.**
8. **Determine a method to scientifically evaluate and determine performance of both systems.**
9. **Accurately measure performance of the two systems.**
10. **Compare and contrast between the findings for both systems to determine improvements or failings.**
11. **Create a recommendation regarding the real-world implementation of containerisation in this use case.**

4 Skills

This is where you can cover the skills you have relevant to the project and the new skills you are going to acquire during the project.

1. Advanced Operating Systems 1, see module KF5004.
2. Computer Networking Experience.
3. Computer Technology, see module KF4004.
4. Virtual machine configuration.
5. LAMP topology experience.
6. Docker configuration.

5 Resources

I will require access to Virtual Machine infrastructure in order to do build my topology and compare virtual machine performance. I will also require access to a machine that has the same hardware and system resources as those shared by the virtual machines, as I will need to perform my containerisation tests on the same hardware in order to effectively control that variable.

5.1 Hardware

I shouldn't require any hardware as long as I can get remote access to the virtual machine infrastructure in a way that allows me to use the same resources for containerisation (as mentioned above).

As contingency, if I cant access this infrastructure, I will still need some way of creating and managing virtualisation. This might be possible on my own hardware, though I'd rather use university infrastructure as this should be more reliable and accessible.

5.2 Software

I will need to install Docker, and will need access to benchmarking suites, but these are free and/or open-source.

6 Structure and Contents of the Report

Below I will set out the chapters that I will most likely have in my final report.

6.1 Report Structure

Abstract & Introduction Here I will set out the background for my project and the reasoning behind the work I will be doing. The Abstract will also summarise the work that has been done along with my findings, subsequent recommendations and conclusions.

What is virtualisation & containerisation An introduction to virtualisation and its current context and usage in today's world of computing. An introduction to containerisation, what it is, how it works, and how it differs to virtualisation. This will form part of my literature review.

Current uses of virtualisation An explanation of modern uses of virtual machines. I will emphasis virtual servers here, and the reasons they are used.

Current uses of containerisation. An explanation of modern usage of containers. I will emphasise application development and other, non-main-sever applications here.

Application of containerisation to server infrastructure Introduce the idea of applying containers to the area I study, with the use-case of server infrastructure. Explain what the implications of this could be. Set out a detailed definition of the problem I want to solve/test. Brief explanation of how this could be designed, with reference to other similar studies.

Network Infrastructure Design Here I will lay out the infrastructure that I want to implement for my testing and my reasoning for this, along with references (see proposed work).

How to test and measure the system Justify the creation or use of a particular set of benchmarks that I will use against the system for testing. I will also explain here how I will measure and evaluate the two systems.

Building the Virtual System Here I will build the virtual system to work with the topology laid out in the infrastructure chapter. I will make sure to explain how to do it. I will also provide evidence of the system working.

Building the Container System Similar to the previous chapter, I will build the container based system following the topology laid out in the infrastructure design section. I will explain how this is done. (ie, using Docker). (see proposed work). As per the previous chapter, I will provide evidence of this working in practice.

Testing and gathering data Here I will create a test plan for running benchmarks across the parallel servers in order to compare the like for like running of both systems.

Analysis & Comparison I will briefly explain how the data shown was required as an introduction to what the data shows. This is where I will explain in detail the actual results, and what they mean, I won't infer here.

Evaluation of the system A technical evaluation of the system I created. I will highlight and strengths, and failings in my ability to meet the requirements of my implementation. I will also ensure to evidence this accordingly.

Evaluation of the project process Here I will evaluate my own learning, and explain what I would do differently about my approach, should I go on to do the same, or a similar, project in the future. My objectives should be assessed as per the marking scheme laid out in the project handbook.

Conclusion and Recommendations I will first create a balanced conclusion, considering the main points mentioned in my evaluation sections. This should cover the direction the work went in, presenting key findings as a way of getting my opinion on the project across. I will present my opinions on the usage of containers in practical setting (ie, whether I think they should be used instead of virtualisation; where they would/could be used) Furthermore, as my work should have practical implications on enterprise and organisations that use virtualisation, I will aim to create recommendations of further work in order to push this area further in the future.

6.2 List of Appendices

My appendices will first include my TOR, Ethics form and Risk Assessment.

Some of the network design documentation may be included if too large for the main body, along with the configurations from the various servers.

I should also include the full and direct results of my benchmarks, as these will almost certainly be too large to display in my main text. (I will instead pull the direct numbers from the benchmarks and reference to the related Appendix).

7 Marking Scheme

The marking scheme sets out what criteria we are going to use for the project.

Project Type General Computing.

Project Report *Analysis Chapters*

- What is virtualisation & containerisation.
- Current uses of virtualisation.
- Current uses of containerisation.
- Application of containerisation to server infrastructure.

Synthesis Chapters

- Network Infrastructure Design.
- How to test and measure the system.
- Building the Virtual System.
- Testing and gathering data.
- Analysis & Comparison.

Evaluation Chapters

- Evaluation of the system.
- Evaluation of the project process.
- Conclusion and Recommendations.

Product The 'product' includes the Network Infrastructure design, the configurations across both systems, the virtual system and the container system themselves.

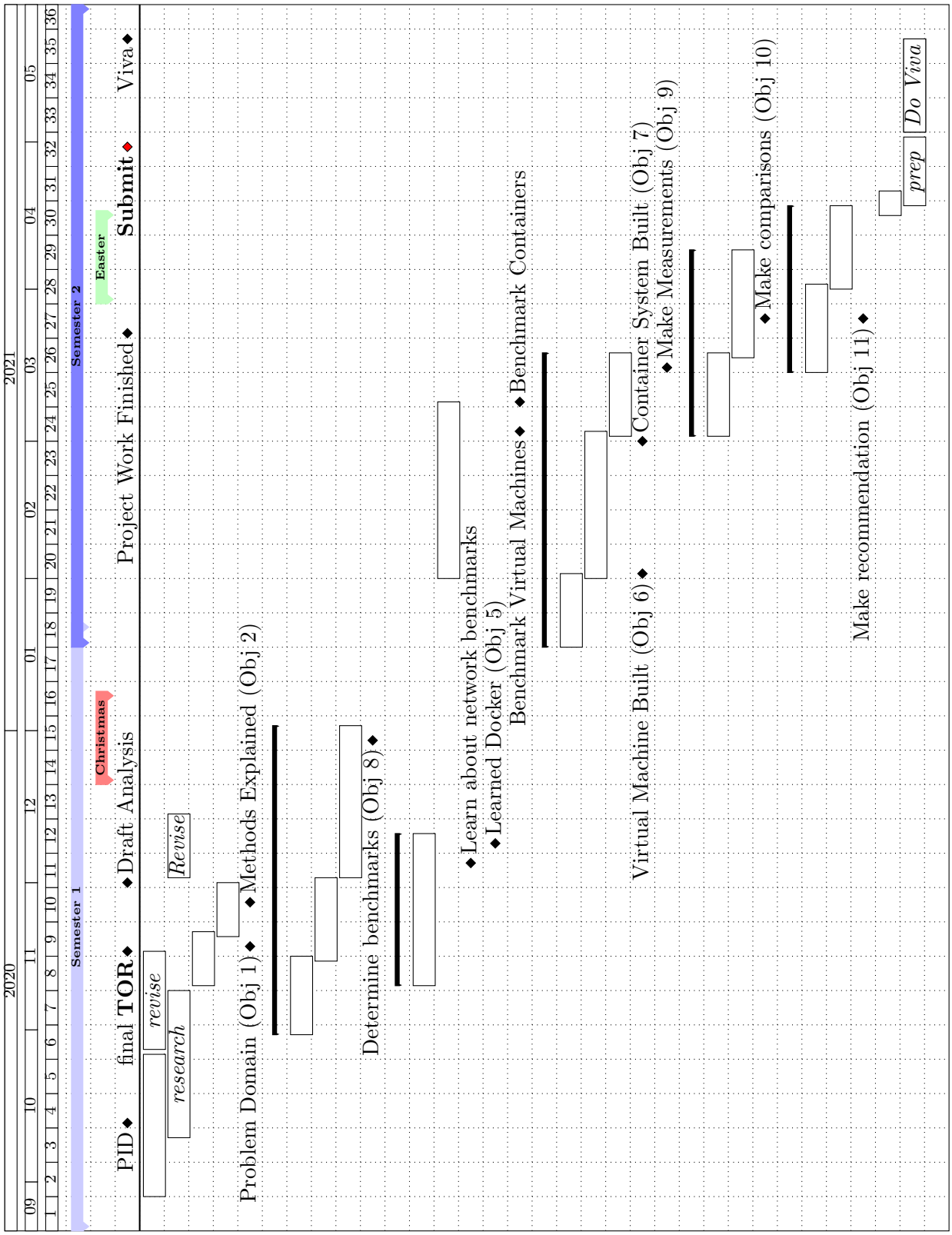
Fitness for Purpose

- Meet requirements identified.
- Application to real world infrastructure.
- Whether the headless servers can communicate effectively on both systems.
- Whether both systems achieved the same goals and outcomes. They should have the same logical topology.

Build Quality

- Requirements specification and analysis.
- Quality of the infrastructure design.
- Configuration quality.
- Benchmarking plan quality.

8 Project Plan



9 Bibliography

R. Dua, A. R. Raja, and D. Kakadia. Virtualization vs containerization to support paas. In *2014 IEEE International Conference on Cloud Engineering*, pages 610–614, 2014. doi: 10.1109/IC2E.2014.41.

A. M. Joy. Performance comparison between linux containers and virtual machines. In *2015 International Conference on Advances in Computer Engineering and Applications*, pages 342–346, 2015. doi: 10.1109/ICACEA.2015.7164727.

Emerson Pugh. *Building IBM : shaping an industry and its technology*. MIT Press, Cambridge, Mass, 1995. ISBN 9780262161473.

451 Research. Application containers will be a $2.7bn market by 2020, 2017. URL `https://451research.com/images/Marketing/press_releases/Application-container-market-will-reach-2-7bn-in-2020_final_graphic.pdf`.

J. Watada, A. Roy, R. Kadikar, H. Pham, and B. Xu. Emerging trends, techniques and open issues of containerization: A review. *IEEE Access*, 7:152443–152472, 2019. doi: 10.1109/ACCESS.2019.2945930.