The evolving grid paradigm and code "tuning" for modern architectures- are the two mutually exclusive?

Robin Long and Roger Jones
Lancaster University

1. Introduction

With the data output from the LHC increasing, many of the LHC experiments have made significant improvements to their code to take advantage of modern CPU architecture and advanced features. With the grid environment changing to heavily include virtualisation and cloud services, we look at whether these two systems can be compatible, or whether improvements in code are lost through virtualisation. This is done by comparing the runtime speed improvements achieved in more recent versions of ATLAS code and seeing if these improvements hold up on various grid paradigms.

2. Containers vs Virtualisation

Traditional virtualisation requires the whole operating system and hardware to be virtualised. Using containers, individual applications can be separated from each other and "virtual machines" can be created but without the overhead of traditional setups.

Containers allow the running of individual applications and their dependencies. In the case of docker, these then run on top of the docker services. Only the userland and applications are virtualised which allows significant overhead savings. This is detailed in figure 1.

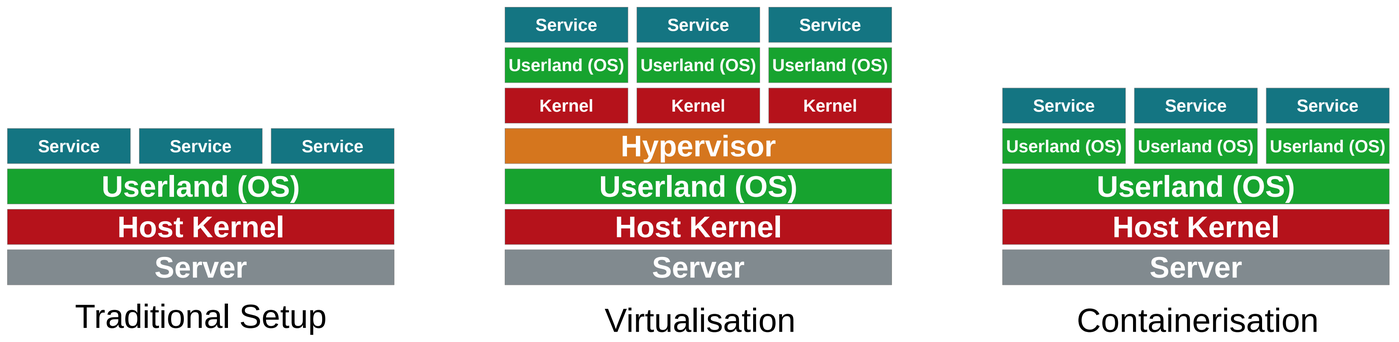


Figure 1: A graphical representation of how bare metal, virtualisation (kvm), and containerisation (docker) works. To run a service on a bare metal machine, we need the kernel, userland (libraries that interact with the kernel), and the software we want to run. When we create a virtual machines, we need to virtualise all of this. Using containers, we only need to virtualise the userland and software as the containers userland interacts with the host kernel.

3. The Analysis

Previous analyses have focussed on benchmarking of nodes. However it is important to test real world code, especially as the software is beginning to diverge away from the benchmarking methods. We specifically focus on the ATLAS software stack, which has had major changes made to its code, and has also changed the libraries that it relies on. The changes studied have resulted in a three fold speed up in run time.

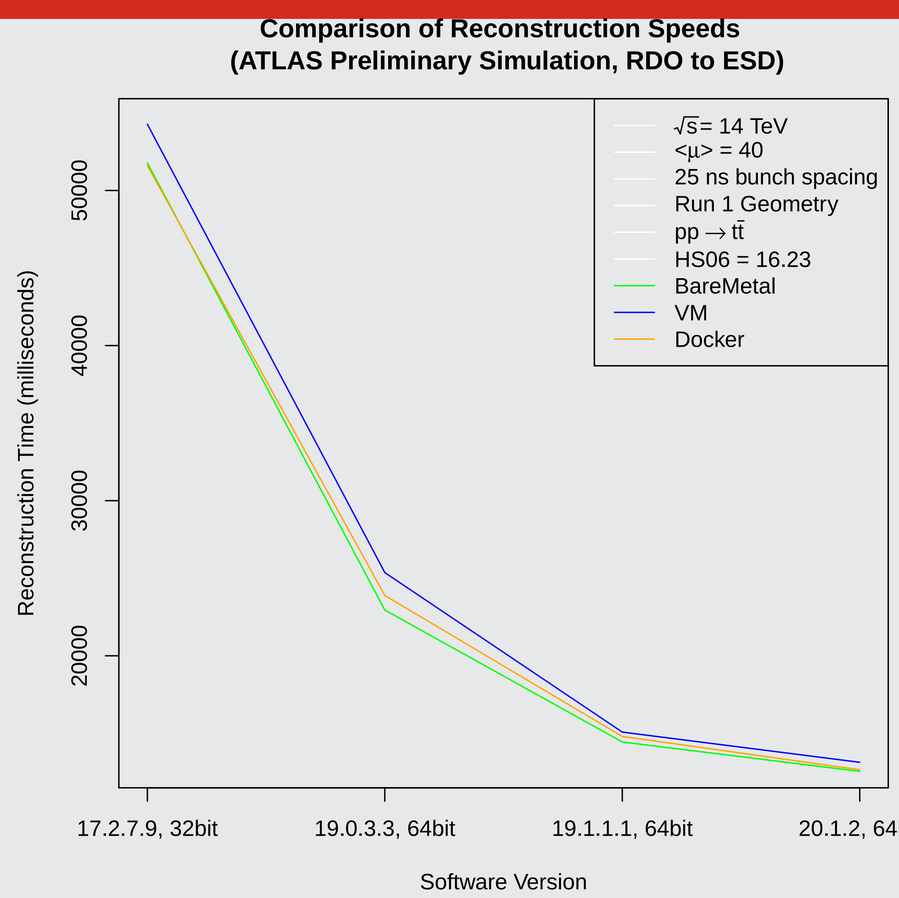
In this analysis we look at three different grid node setups (all using Scientific Linux 6): Bare Metal, Full virtualisation using kvm, and containerisation using Docker. By running four different releases of the ATLAS software on the 3 different node configurations, we hope to see whether the performance benefits brought about from the software changes are maintained, and whether the two virtualisation techniques have a significant impact on runtime speeds.

4. Results

Two things are immediatly noticable from this plot. Firstly, all three methods (Baremetal, Virtualisation, and containerisation) have very similar levels of improvement as the software version is changed. Secondly, the time taken to reconstruct individual events is comparable for all three methods across the 4 software versions.

As the software versions are improved dramatic decreases are seen in the time taken to reconstruct each event (in milliseconds). These improvements are seen across all three paradigms. Whilst virtualisation and containerisation are slower, there is no noticable change in the rate of progress as the software is improved.

For Docker and Bare Metal there is a significant amount of overlap in the results. Any performance loss in Docker is within the variation seen in the results. In some cases Docker appears to be faster than Bare Metal, but more repetitions removes this artefact and shows that the two are almost identical.



5. Conclusion

Virtualisation and containerisation are shown to have no significant effect on total reconstruction time. For version 17.2.7.9 of the software, Docker takes on average 0.17 (0.32%) seconds less to reconstruct an event compared to Bare Metal, whereas virtualisation takes 2.5 seconds (4.81%) longer. By version 20.2.1.Y this has changed to 0.1 (0.85%) and 0.58 seconds (4.66%) longer respectively.

The improvements being made to ATLAS software are not harmed by any of the considered grid paradigms. Furthermore, the software is not hugly affected by differing paradigms to begin with.

Acknowledgements

The authors would like to thank the ATLAS collaboration for allowing use of the software and datasets for this analysis. Furthermore a special thanks is extended to Antonio Limosani, whose help and scripts made this analysis possible.

http://www.hep.lancs.ac.uk/∼rel
r dot long at lancaster dot ac dot uk