Could the Topology of Virtual Processors Affect the Performance of a BSD-family OS Running in a VM?

David Beserra*, Marc Espie, Léo Tomasimo, Hector de Poncins, Hadrian Lacombe, Tomas Vondracek
École Pour l'Informatique et les Techniques Avancées
Paris, France
*david.beserra@epita.fr, marc.espie.openbsd@gmail.com

Jean Araujo
Universidade Federal do Agreste de Pernambuco, UFAPE
Garanhuns, Brasil
jean.teixeira@ufape.edu.br

***Abstract* — Virtual machines are an essential technology in distributed and pervasive systems. One of its configurable parameters is the topology of the virtual processing system, which can potentially impact its performance. In this work, we verify how different virtual processing topologies affect the performance of VMs running BSD OSes. We conclude that in some types of application the topology does not affect the VM performance, while in others it does, and that the performance impact also depends on the OS adopted by the VM.**

Keywords - Distributed Systems; OpenBSD; FreeBSD; Virtualization; Processor Topology; Performance Evaluation.

I. INTRODUCTION

Virtualization is a key technology for many applications and computing paradigms, particularly those related to distributed systems, such as Pervasive Computing [1], [2] and High Performance Computing [3]. One of its advantages is the flexibility of Virtual Machines (VMs) in terms of configuration [4]. For instance, it is possible to configure the amount and type of CPUs used by the VM, as well as the amount of RAM and many network-related parameters [3], [5].

One of the configurable parameters of VMs is the topology of the processing system, i.e., the distribution of cores among processors [6]. As the virtual processor topology can differ from the topology of the physical machine, it may impact the VM performance. Therefore, it is important to investigate whether the virtual processor topology affects the performance of the VM in terms of a measurable metric, such as the number of Floating Point Operations Per Second (FLOPS) [7].

However, to ensure that any possible performance issues arising from different virtual processor topologies are not due to chance or to problems linked to a single OS, it is necessary to conduct tests with different OSes. To this end, we have chosen to examine how the topology of virtual processors affects the performance of VMs running different OS from the BSD family. We focus on BSD distributions, such as OpenBSD, because they are fully open source and place more emphasis on system security than Linux, and security is a critical aspect of current distributed systems. Therefore, in this study, we propose to investigate whether the virtual processor topology adopted for a VM impacts its performance, and if so, whether this impact is tied to the specific choice of OS.

This work is structured as follows: Section II describes and analyzes some selected related works. Section III presents our main goals and the methodology used in our experiments, then we present and discuss the results in Section IV. Finally, Section V presents the conclusions obtained from this study and our plans for future work.

II. RELATED WORKS

Studies that explore the relationship between processing topology and performance are not new to computer science. For example, [8] evaluates the effects of Sub-NUMA Clustering (SNC) on memory latency and bandwidth on Intel processors with NUMA topologies, while [9] evaluates the use of graphs for mapping tasks to physical processors, taking their topology into account. Finally, [10] investigates how to map application processes onto manycore processors using an active search framework. However, these works focus on physical processors and sometimes interpret "processor topology" in a broad sense, such as considering a cluster or a grid as a single processor [11]. While these works are interesting, they do not answer the questions addressed in this paper.

More recent works have evaluated the performance of virtualized environments using groups of VMs [3], [12] or containers [5] running on the same host and organized according to some topology. While these studies found that the topology of the environment can affect performance in some cases [4], in others it does not [5]. Moreover, they did not consider the effects of processor topology on performance, nor did they examine multiple operating systems. Jang et al. [13] investigates the scheduling of virtual CPUs to tasks, but does not consider the possible relationships between the virtual processor topology and the VMs performance.

Regarding systems in the BSD family, some studies have investigated their performance-related issues. For example, Vavrenyuk et al. [14] evaluates the performance of OpenBSD for HPC tasks on Single Board Computers (SBCs), while Mitran et al. [15] investigates the performance of FreeBSD virtualized nodes in an OpenStack environment. However, these studies either do not take processing topology into account, or are limited to a single type of BSD OS.

To the best of our knowledge, no previous study has used VMs to investigate whether different processor topologies with equal processing cores and shared memory affect the performance of BSD systems. In this paper, we propose to fill this gap in the scientific literature. The experimental design for this study is described in more detail in the next section.

2023 18th Iberian Conference on Information Systems and Technologies (CISTI)
20 – 23 June 2023, Aveiro, Portugal
ISBN: 978-989-33-4792-8

III. EXPERIMENTAL DESIGN

The main goal of this work is to check whether the virtual processor topology adopted in a VM affects its performance. In order to reach this goal, we evaluate the performance of some of the main BSD systems running on a virtual machine, with the following specific questions :

- Does the virtual processor topology affect the performance of **cooperative** CPU-bound applications performance?
- Does the virtual processor topology affect the performance of **concurrent** CPU-bound applications performance?
- Does the virtual processor topology affect the interprocess communication performance, considering intranode communications?.

A. Performance Metrics and Benchmark Adopted

In order to evaluate each one of the targeted goals we need to associate these goals with measurable metrics. In this work we have chosen the following metrics:

- Amount of floating-point operations per second that can be performed by a processor in a given time (FLOPS);
- Communication latency between MPI process pairs, measured in nanoseconds;
- Communication bandwidth between MPI process pairs, measured in Megabits per second (Mbps).

Each of these metrics is associated with one of our research goals and measured using a specific benchmark. For instance, metric 1 was used to measure processing capacity of the system being evaluated when all processors are cooperating to solve a single problem, and it associated with the question 1. To measure this metric, we used the **High Performance Linpack (HPL)** benchmark. This benchmark measures the amount of floating-point operations per second (Flops) performed by a computational system while solving a dense linear equations system [7]. We chose HPL because it is the default benchmark for measuring the processing capacity of parallel computers, including those that figure in the TOP500 ranking of supercomputers [16].

To run the HPL benchmark adequately, we need to specify the order N of the linear system to be solved, the topology of the processors grid ($P \times Q$) and other configurable parameters. Since the N size is crucial for achieving a good performance with HPL, we used **Tuning HPL**, a Web mechanism provided by *Advanced Clusters Inc.* that automatically generate configurations to improve our experiment setup.

As metric 1 is also associated with question 2, we also used HPL as a measurement instrument, but we ran several concurrent instances of HPL competing for resources, then we measured the execution time of each one. Besides, the order N of the linear system to be solved by each instance is smaller than in the previous experiment, in order to have enough RAM to execute all instances. Finally, in this case the value used by the processor grid is always 1. Thus, we can see the effect of the competition of programs for resources and check whether the operating system scheduler distributes processing time fairly and whether the topology affects this scheduling.

Metrics 2 and 3 were used to measure the inter-process communication capacity in the systems evaluated. These metrics are associated with question 3 mentioned above. To measure these, we used the **Network Protocol Independent Performance Evaluator (NetPIPE)** benchmark [17]: this benchmark monitors communication overheads using different protocols like TCP, UDP and MPI. It performs simple ping-pong tests, sending and receiving messages of increasing size between a couple of processes, either across a cluster connected by an Ethernet network or within a single multicore system (our case).

A. Infrastructure (Physical and Virtual)

The experiments were conducted on a computer fitted with an Intel 11th Generation Core i5-1135G7 processor. This processor contains 8 processor cores running at 2.5 GHz. The computer also possesses 8 GB of DDR4 RAM memory running at 3.2 GHz. As secondary memory, it uses an SSD with 256 GB available. The OS installed on this computer is Kali GNU/Linux Rolling x86 64 6.0.0-kali6-amd64.

This computer was used to host virtual machines implemented with KVM (Kernel-based Virtual Machine), a type-1 hypervisor that allows configuration of the processor topology of the VMs. We chose KVM instead of type-2 hypervisors like QEMU and Virtualbox due to performance issues. We also excluded the Xen hypervisor even though Xen is also a type-1 hypervisor because KVM is easy to install and operate and has similar performance as Xen.

OS-level virtualization solutions such as LXC and Docker were excluded because they assume that the same kernel is shared between several containers, thus limiting tests to other linux systems instead of our intended BSD targets. Also, from a security standpoint, such solutions are known to have issues with resource isolation between different containers.

A. Experiences Performed

To reach the goals targeted on this research, we evaluated the performance of different environments. Each environment is composed of a single VM running one of the OS described in Table I, and with 6 processing cores organised according to one of the topologies in Table II. Each topology corresponds to a certain number of processors and processing cores associated to each processor.

TABLE I. OPERATING SYSTEMS USED IN THE EXPERIMENTS

OS	Version
NetBSD	12.4
FreeBSD	9.2
OpenBSD	7.2
DragonFly BSD	6.2.2

Each environment has 4 GB of RAM, regardless of the topology adopted. In this way, 2 processing cores and 4 GB of RAM are always left available for the host hosting the VMs,

ISBN: 978-989-33-4792-8

thus reducing the risk of overheads on the VMs caused by resources starvation on the host.

TABLE II. PROCESSOR TOPOLOGIES ADOPTED

Topology	Processors	Cores/Processor
1proc-6core	1	6
2proc-3core	2	3
3proc-2core	3	2
6proc-2core	6	1

All benchmarks were executed 32 times, for all tests, scenarios and conditions. In all tests, the two measurements with highest and lowest value were excluded as outliers; we computed the average and standard deviation from the remaining 30 measurements. The next subsections explain in more details the experimental procedures for each target goal.

1) Experiment 1 : system's performance on cooperative cpu-bound tasks: To evaluate the processing performance of the system when all its processing cores are working together in a cooperative way we instantiated only one instance of the HPL benchmark. This instance generates 6 processes and solves a linear system where the order N is 18816. Each one of these processes is attached to a processing core. We repeat this experiment for each topology described in the Table II. We assert that, if the topology does not matter, then the performance obtained on the tests will always be roughly the same. If the topology matters, we assert that distinct performances will be observed for each topology evaluated.

1) Experiment 2 : system's performance on competitive cpu-bound tasks: This experiment is divided in three steps. In the first step, we launched a single HPL instance, associated with one 1 process. In the second step, we launched 3 instances of the HPL benchmark, each one associated with one process and solving a linear system where $N = 8064$. In theory, only half of the processors available will be busy, and the execution time of each instance should be roughly equal for each environment, independent of the processor topology adopted. In the third step, we launched 6 instances of the HPL benchmark, each one associated with a process and, in theory, running on different processing cores. Here, we suppose that the competition between light processes from guest and host OS could have a slight impact on the performance of each HPL instance. However, if the performance differences are too disparate, then the processor topology has to be one of the factors affecting performance.

1) Experiment 3: inter-process latency and bandwidth: In this experiment we used NetPIPE to verify if the processor topology has some impact on the performance in intra-node inter-process communications. Here, we suppose that the communication latency can be highly variable as a function of the processor topology, but the communication bandwidth will not necessarily be affected.

IV. RESULTS

This section presents the results of the research conducted. The results are organized according to the experiments performed. Subsection IV-A presents the results related to Experiment 1, while Subsection IV-B presents the results of Experiment 2, and finally, Subsection IV-C presents the results of Experiment 3.

A. Experiment 1: Results and Analysis

Figure 1 shows the results obtained when the HPL benchmark was executed in cooperative mode. In general, with regard to the standard deviation, it can be seen that the processor topology does not have a significant impact on the computing power of the VMs running under OpenBSD and DragonFly BSD, but it has some impact under FreeBSD when it is under the 6proc-1core topology. Moreover, the VM running NetBSD presented unexpected behavior, where topologies with more processors presented higher performance than those with fewer processors. Further testing may be necessary to explain these results.

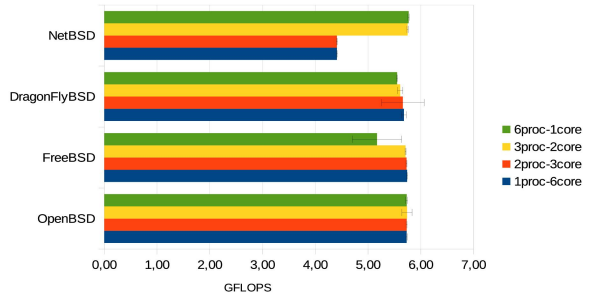

Figure 1. HPL cooperative.

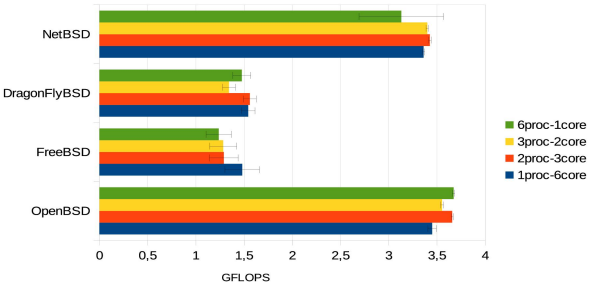

Figure 2. HPL competitive (1 instance running).

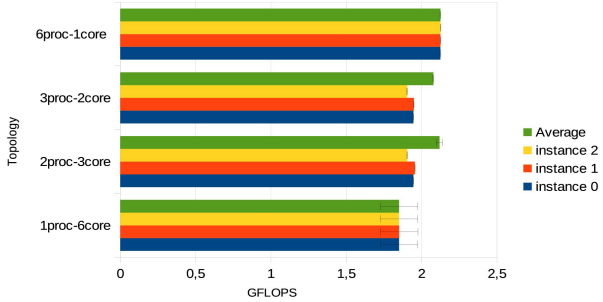

Figure 3. HPL competitive (3 instances running) - OpenBSD.

2023 18th Iberian Conference on Information Systems and Technologies (CISTI)
20 – 23 June 2023, Aveiro, Portugal
ISBN: 978-989-33-4792-8

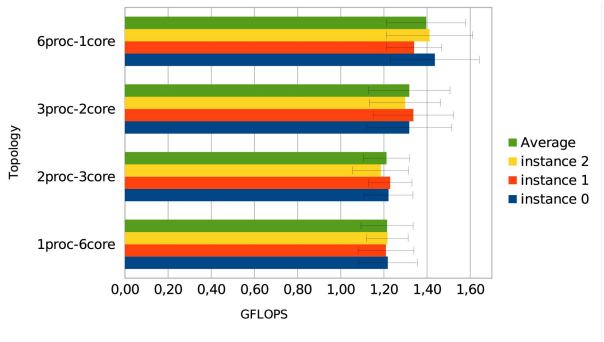

Figure 4. HPL competitive (3 instances running) - FreeBSD

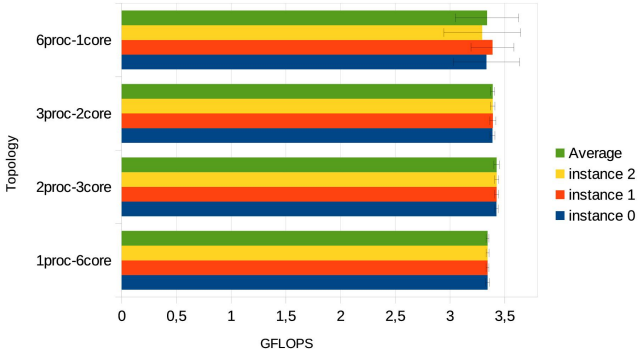

Figure 5. HPL competitive (3 instances running) - NetBSD.

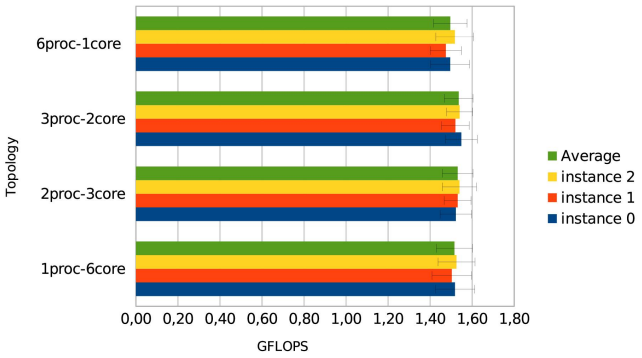

Figure 6. HPL competitive (3 instances running) – DragonFly BSD.

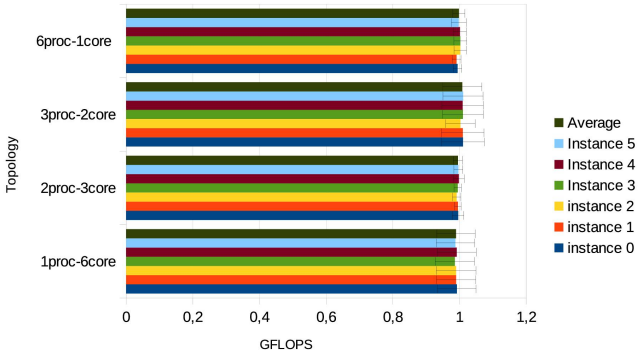

Figure 7. HPL competitive (6 instances running) - OpenBSD.

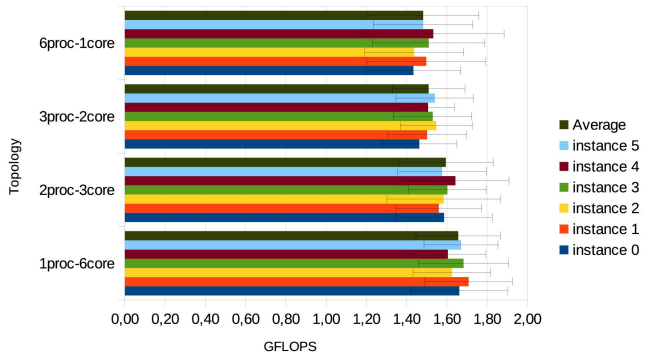

Figure 8. HPL competitive (6 instances running) - FreeBSD.

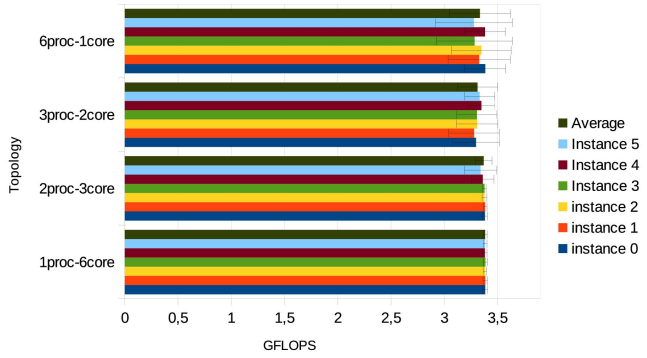

Figure 9. HPL competitive (6 instances running) - NetBSD.

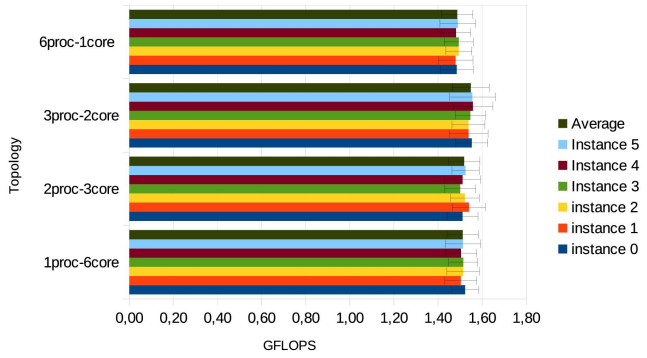

Figure 10. HPL competitive (6 instances running) – DragonFly BSD.

A. Experiment 2: Results and Analysis

Figure 2 presents the results of running a standalone instance of the HPL benchmark on a single processor. In some cases, the processor topology has an impact on computing performance. For example, while the VM running OpenBSD performed almost equally for all topologies, the performance of the VM running FreeBSD clearly decreased when the number of v-CPUs increased. Next, DragonFly BSD and NetBSD also presented similar performance for most topologies. However, DragonFly BSD presented an unexpected performance reduction for the 3proc-2core topology, and NetBSD presented higher performance variations under the 6proc-1core topology.

Figures 3-6 shows the results of Experiment 2b, with more independent instances running at the same time. It is easier to

2023 18th Iberian Conference on Information Systems and Technologies (CISTI)
20 – 23 June 2023, Aveiro, Portugal
ISBN: 978-989-33-4792-8

see the effects of different processor topologies on different BSD systems. For instance, OpenBSD and FreeBSD presented better performance for concurrent applications when using the topology 6proc-1core. It means that, for these systems, it may be more interesting to deploy VMs with multiple processors but fewer cores per processor if the main goal is to execute independent CPU-intensive service modules. When considering this experiment, it is not possible to say that the processor topology affects its performance because the average performance of each HPL instance ran is nearly identical for all topologies evaluated. Finally, NetBSD presents the same behavior as in the previous experiment, where the measured performance is almost the same for almost all topologies, except for the 6proc-1core topology.

The results obtained for Experiment 2c (shown in Figures 7-10) follow the same pattern as the previous experiments. The VM running OpenBSD presents a stable behavior and is not affected by the virtual processor topology. The same can be said about the VMs running DragonFly BSD and FreeBSD, but with higher performance variations because they are running more HPL instances. In the particular case of FreeBSD, we can also see that performance variability increases as a function of the number of v-CPUs adopted in the processor topology, and we can conclude that the adopted topology impacts the performance of VMs under this system. Finally, NetBSD follows the same performance pattern observed previously, and we can conclude that the adopted topology impacts the performance of VMs under this system.

A. Experiment 3: Results and Analysis

In this section, we present the results obtained regarding the inter-process communication performance in each environment and for each processor topology adopted. The results obtained for Experiment 3 are shown in Figures 11-18 below. From these figures, we can see that the bandwidth performance of the inter-process communication was not affected by the processor topology. The same can be said regarding the latency in the communication between processes.

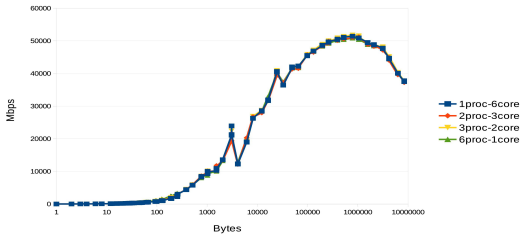

Figure 11. Interprocess communication – bandwidth - OpenBSD.

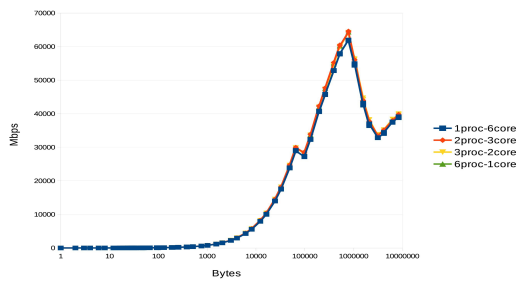

Figure 12. Interprocess communication – bandwidth - FreeBSD.

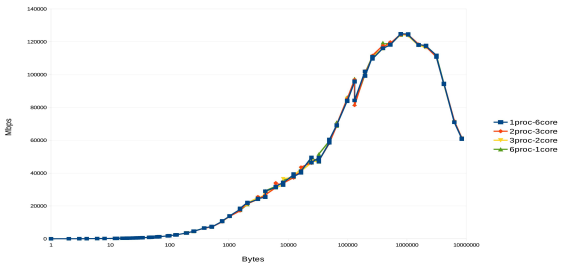

Figure 13. Interprocess communication – bandwidth - NetBSD.

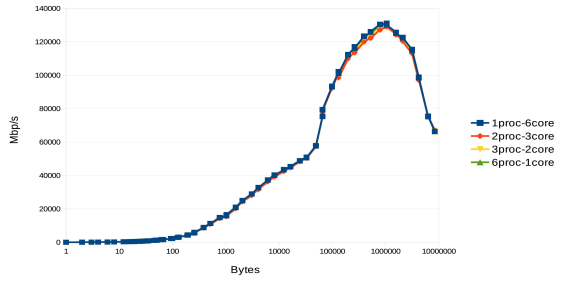

Figure 14. Interprocess communication – bandwidth – DragonFly BSD.

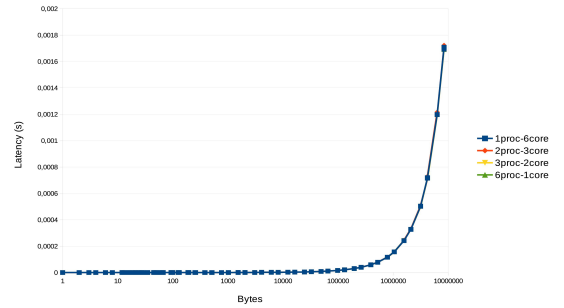

Figure 15. Interprocess communication – latency – OpenBSD.

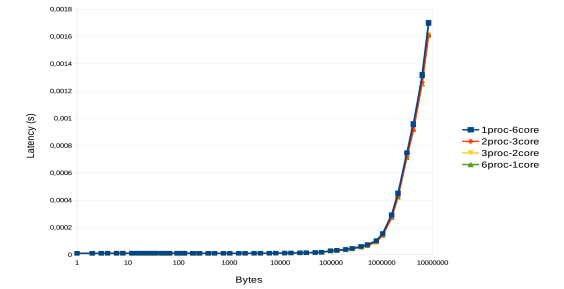

Figure 16. Interprocess communication – latency - FreeBSD.

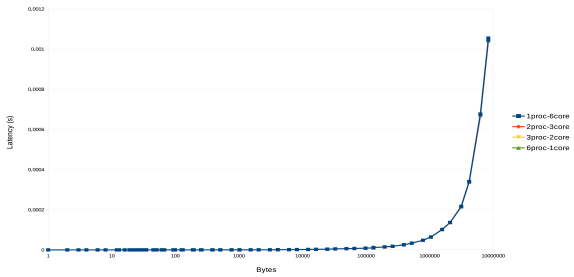

Figure 17. Interprocess communication – latency - NetBSD.

2023 18th Iberian Conference on Information Systems and Technologies (CISTI)
20 – 23 June 2023, Aveiro, Portugal
ISBN: 978-989-33-4792-8

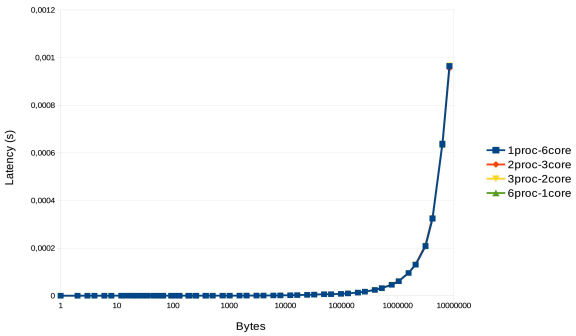

Figure 18. Interprocess communication – latency – DragonFly BSD.

V. Conclusions

Based on the results obtained in this study, it was possible to evaluate the impact of different processor topologies on the performance of VMs running different BSD systems. In general, it was observed that the processor topology does not have a significant impact on the computing power of VMs running OpenBSD and DragonFly BSD. However, it does have an impact on the performance of VMs running FreeBSD, especially for topologies with more v-CPUs. Finally, the VM running NetBSD presented some unexpected behavior that requires further investigation.

When considering interprocess communication performance, the topology of the processor did not have a significant impact on the bandwidth and latency of communication between processes. Therefore, we can conclude that different BSD systems may react differently to different processor topologies, and this should be taken into account when deploying VMs. It is recommended that system administrators conduct performance tests on different topologies to determine the best configuration for their specific needs.

As for future work, we suggest further investigation of the unexpected behavior observed for the VM running NetBSD. Additionally, more experiments could be conducted with different benchmarks and workloads to better evaluate the impact of processor topology on VM performance. Finally, it would be interesting to study the impact of other virtualization technologies, such as containers, on BSD systems.

References

[1] L. Rudolph, “A virtualization infrastructure that supports pervasive computing,” IEEE Pervasive Computing, vol. 8, no. 4, pp. 8–13, 2009.

[2] J. Martins, J. Alves, J. Cabral, A. Tavares, and S. Pinto, “µ rtzvisor: a secure and safe real-time hypervisor,” Electronics, vol. 6, no. 4, p. 93, 2017.

[3] D. Beserra, E. D. Moreno, P. Takako Endo, J. Barreto, D. Sadok, and S. Fernandes, “Performance analysis of lxc for hpc environments,” in Complex, Intelligent, and Software Intensive Systems (CISIS), 2015 Ninth International Conference on. IEEE, 2015, pp. 358–363.

[4] D. Beserra, F. Oliveira, J. Araujo, F. Fernandes, A. Ara´ujo, P. Endo, P. Maciel, and E. D. Moreno, “Performance evaluation of hypervisors for hpc applications,” in Systems, Man and Cybernetics, IEEE International Conference On. IEEE, 2015, pp. 846–851.

[5] D. Beserra, M. K. Pinheiro, C. Souveyet, L. A. Steffenel, and E. D. Moreno, “Performance evaluation of os-level virtualization solutions for hpc purposes on soc-based systems,” in 2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA). IEEE, 2017, pp. 363–370.

[6] I. Corporation, “Intel 64 architecture processor topology enumeration - whitepaper,” INTEL, Tech. Rep., 2018.

[7] P. Luszczek, E. Meek, S. Moore, D. Terpstra, V. M. Weaver, and J. Dongarra, “Evaluation of the hpc challenge benchmarks in virtualized environments,” in Euro-Par 2011: Parallel Processing Workshops. Springer, 2012, pp. 436–445.

[8] S. K. Mohapatra et al., “Authentication of sub-numa clustering effect on intel skylake for memory latency and bandwidth,” Turkish Journal of Computer and Mathematics Education (TURCOMAT), vol. 12, no. 11, pp. 195–204, 2021.

[9] S. Kalyur and G. Nagaraja, “Evaluation of graph algorithms for mapping tasks to processors,” in 2nd EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing: BDCC 2019. Springer, 2021, pp. 423–448.

[10] R. Sambangi, A. S. Pandey, K. Manna, S. Mahapatra, and S. Chatterjee, “Application mapping onto manycore processor architectures using active search framework,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2023.

[11] Y. He, D. Yu, Y. Deng, and J. Lei, “An efficient topology processor for distribution systems,” in 2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No. 01CH37194), vol. 2. IEEE, 2001, pp. 824–829.

[12] D. Beserra, E. D. Moreno, P. T. Endo, J. Barreto, S. Fernandes, and D. Sadok, “Performance analysis of linux containers for high perfor- mance computing applications,” International Journal of Grid and Utility Computing, vol. 8, no. 4, pp. 321–329, 2017.

[13] J. Jang, J. Jung, and J. Hong, “An efficient virtual cpu scheduling in cloud computing,” Soft Computing, vol. 24, no. 8, pp. 5987–5997, 2020.

[14] A. Vavrenyuk, V. Makarov, V. Pryakhin, M. Pavlov, and A. Vasileva, “Performance evaluation of a cluster computing system running openbsd based on single-board computers,” in Advanced Technologies in Robotics and Intelligent Systems: Proceedings of ITR 2019. Springer, 2020, pp. 121–126.

[15] A. Mitran, M.-E. Mih˘ailescu, D. Mihai, S. Weisz, M. Carabas, and N. T˘apus, “Freebsd as a compute node in openstack,” in 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP). IEEE, 2020, pp. 547–554.

[16] J. Napper and P. Bientinesi, “Can cloud computing reach the top500?” in Combined Workshops on UnConventional High Performance Computing Workshop Plus Memory Access Workshop, ser. UCHPC-MAW ’09. New York, NY, USA: ACM, 2009, pp. 17–20.

[17] Q. O. Snell, A. R. Mikler, and J. L. Gustafson, “Netpipe: A network protocol independent performance evaluator,” in IASTED international conference on intelligent information management and systems, vol. 6. (Washington, DC, USA),, 1996, p. 49.

ISBN: 978-989-33-4792-8