DOI: 10.2507/30th.daaam.proceedings.050

INFLUENCE OF VIRTUAL STORAGE CONTROLLER TYPE ON MICROSOFT SQL SERVER 2019 PERFORMANCE

Vedran Dakic

This Publication has to be referred as: Dakic, V[edran] (2019). Influence of Virtual Storage Controller Type on Microsoft SQL Server 2019 Performance, Proceedings of the 30th DAAAM International Symposium, pp.0370-0375, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-22-8, ISSN 1726-9679, Vienna, Austria
DOI: 10.2507/30th.daaam.proceedings.050

Abstract

Influence of different virtual storage controller types on database performance when using VMware virtualization technology has long been debated, but not analyzed properly. In this paper, we're going to analyze the impact of different storage controller types on Microsoft SQL Server 2019 database performance on VMware's vSphere 6.0 hypervisor and try to reach a conclusion as to what's the correct way to design server and virtual machine infrastructure for optimum database performance when using a regular, on-disk database. We will also analyze potential problems that might arise from situations when we migrate from other virtual disk types to paravirtual.

Keywords: SQL; performance; storage controller; database; paravirtual

1. Introduction

When we use Microsoft SQL database from a client perspective, we're usually not concerned with the virtual machine configuration aspect - we kind of treat it more like an afterthought. Often times, end-users will complain that their applications (which use a SQL backend) is slow, inconsistent, and unpredictable in terms of speed and service levels. Then, virtualization administrators usually try to convince users and their managers that the problem lies in the fact that the virtual machine doesn't use enough virtual memory (which can actually be a part of the problem), and/or that virtual machine doesn't have enough CPU power (which is rarely the problem). So, these types of problems often feed on themselves, by creating users that are unhappy with database performance, administrators that often look for a solution in the wrong place, managers who are unhappy with the fact that additional investments are often presented as a solution, and... we end up going in circles, without actually analyzing what the problem might be. Saying that "storage is the problem", "memory is the problem" or "lack of CPU resources is the problem" are often just a generalized way of saying "we didn't really check, but we're guessing that this is the problem".

Problem with this way of thinking is not only the lack of the engineering approach, it's a general design problem. People tend to just create virtual machines with default option and not bother with the available additional options, and that can significantly impact performance of virtual machines. To put things into larger context, when we're designing virtual machines which will run detabases, we're usually more concerned with questions like "how much memory will we give to our virtual machine", or "how many CPUs should we give to our virtual machine".

Sometimes we just forget that – for databases – constant level of disk performance in terms of throughput, read and write latencies is just as important. If we're not careful with the process of selection virtual storage controller types, we might end up in a situation in which virtual machines running databases offer levels of service that is more random then constant. And we should always strive to avoid that.

This prompted us to research the subject of correct design, with specific scope being choosing a correct virtual storage controller - either out of the box - before the virtual machine is deployed, or after the deployment process, and to consider any possible problems or downfalls of doing so.

VMware vSphere Hypervisor supports four different types of virtual storage controllers which we can use from our virtual machines:

- LSI Logic Parallel
- BusLogic Parallel
- LSI Logic SAS
- VMware Paravirtual

Guest operating systems that we install in our virtual machines are a bit choosy when it comes to storage controller support, so - when we design virtual machines, we usually do that by using a New Virtual Machine wizard, which uses our operating system selection to choose a different virtual storage controller type for us. With OLTP (*Online transaction processing*) scenario in mind, if we follow along with the best practices "databases are usually characterized with mostly intensive random writes to disk and sustained CPU utilization during working hours" [1]. This means that there are multiple factors that will influence the performance of our OLTP database:

- CPU utilization from the virtualization host perspective;
- CPU utilization from the virtual machine perspective;
- amount and speed of the memory that we are using for our database virtual machine;
- any kind of SCSI command queueing - on virtual or physical storage adapter or network if we are using traditional shared storage (iSCSI, Fiber Channel, Fiber Channel over Ethernet);
- virtual disk performance, which is closely related to virtual storage controller performance.

First two of these factors are very common, and what we usually do is design our physical and virtual servers so that we have some available resources. "Some customers have established targets for system utilization on hosts running SQL Server, for example, 80 percent CPU utilization, leaving enough headroom for any usage spikes and/or availability" [1].

Third factor - amount and speed of memory that we are using for our database virtual machine - is a relatively straightforward problem to solve. We can just configure our virtual machine so that it has a completely *reserved* memory, which means that it will only use real, RAM memory from the host, instead of using swapping on virtual machine, host or hypervisor level. Furthermore, it will prevent other memory reclamation techniques like TPS (*Transparent Page Sharing*) and memory compression to have any kind of influence on our memory performance. That will directly translate to performance of our SQL database in the virtual machine, as there will not be any spikes in latency which would happen if we had to load contents from a datastore or local disk space.

Fourth factor, which is related to queueing on any physical or virtual storage adapter, is something that we can also plan for and configure, by using features like Storage and Network I/O control. We use Storage I/O control so that we can use automatic measuring of peak throughput percentage or latency to make decisions about "importance" of input/output requests that were issued by virtual machines. That means that we can select which virtual machines are more critical in our environments and keep their performance at a constant level. By using Netw[ork I/O control, we can assign specific amount of available bandwidth from a pool of network adapters available on our host for iSCSI storage access, just as an example. Then, we can use Storage I/O Control exactly like we described earlier on top of Network I/O control - to make sure that our iSCSI storage performance is also kept at a constant level. In practice, that means that, even though queueing could theoretically happen, it's much less likely to happen (or won't happen at all) for our critical virtual machines, which is what we're after.

Fifth factor, which is the basis of our research, is related to virtual storage controller performance, as there are differences in performance between the aforementioned virtual storage controller types. In this research, we will try to find a design framework which will tell us when to use certain virtual storage controller type and why. We have to bear in mind that virtual storage controller selection has other side-effects - for example, on the overall CPU overhead level per host. Some of these virtual storage controllers should have significantly lower overhead, which should also mean better performance and lower latency.

VMware presents the Paravirtual virtual storage controller as the one to use when we need lower latency, better throughput and lower CPU overhead from the host perspective. According to available VMware documentation, "the PVSCSI adapter offers a significant reduction in CPU utilization as well as potentially increased throughput compared to the default virtual storage adapters, and is thus the best choice for environments with very I/O-intensive guest applications" [2].

Our area of interest is - at least partially - related to NUMA locality as well. Specifically, we were very interested to know what kind of influence (if any) will NUMA configuration have on virtual storage controller performance. Having in mind that there's much more context switching needed if we have many NUMA nodes versus a situation in which we have a smaller number of NUMA nodes, this might have an impact on the virtual storage controller performance.

What we wanted to learn with this research and its methodology is:

- is using VMware Paravirtual storage controller makes a big difference and in which cases?
- is it a worthwhile investment of time and money to move from LSI Logic SAS controller to VMware Paravirtual, which is what often happens as administrators hear about this previously mentioned VMware recommendation?
- is there any influence that an (in)correct NUMA configuration might have on the virtual storage controller performance?
- are there any inherit risks in doing so which go beyond the purely performance standpoint?

2. Methodology

Testing methology that we selected for this paper uses a standard AdventureWorks database which is available on GitHub and from Microsoft (https://bit.ly/2zsevpc). We used a HP DL 380 Gen9 server with two E5-2698v3 16-core CPU, 192 GB of memory and IBM DS3524 dual-controller 8Gbit/s Fiber Channel storage. Storage was configured with RAID 10 configuration, consisting of 16 300GB 10.000rpm 2.5" hard drives. Our virtual machine infrastructure was configured with two most commonly used types of storage controllers - LSI Logic SAS and VMware Paravirtual. All virtual disks were stored on the IBM DS3524 storage. Virtual machine with SQL database was configured with best practices in mind - we separated database files from transaction logs. So, when we were testing LSI Logic SAS-based virtual controllers, we added two disks to that controller and used the first one for database files, and the second one for log files. The same was done for testing with VMware Paravirtual virtual storage controller. All virtual disks were provisioned with "Thick Provision Eager Zeroed" policy. We wanted to avoid any kind of performance drop that usually happens if we used other policies (Thick Provision Lazy Zeroed, or Thin Provision). Also, all of the memory for our virtual machine was pre-reserved, so that we don't have any concerns about performance drops because of swapping.

In our testing, we measured on-disk memory database performance by using a very memory-intensive test, a *cross-join* between two tables. You can read more about cross join and this specific test in our other paper, called "Influence of NUMA and memory locality on Microsoft SQL Server 2019 performance" [3]. We've used the same dataset, but a completely different type of database (in-memory versus on-disk), which yielded a set of completely different results. Also, we only used one of the two tests which is directly related to loading database data from disk. Keep in mind that modern server applications are expected to process a huge number of requests simultaneously without noticeable degradation of performance, e.g. response times and throughput [4], which is what we're indirectly measuring in our paper. This has direct influence on service/performance level, which is usually defined by some kind of SLA (Service Level Agreement). SLAs commonly include segments to address: a description of services, performance measurement, problem management, customer duty, warranty, failure recovery, termination of agreement.[5].

By selecting Fibre Channel-based storage, we tried to avoid any bottlenecks that we might have got if we used iSCSI-based storage. For example, a common problem that we see is a switch in the data path into the storage system that is fragmenting frame [6], or just overloaded. Fibre Channel doesn't fragment packages, has no retransmissions, or encapsulation like iSCSI. Therefor, it's much better suited for performing tests like these.

3. Results

Let's check out results table:

Sample size	1 socket x 16 cores	2 socket x 8 cores	4 sockets x 4 cores	8 sockets x 2 cores	16 sockets x 1 cores
500x500 Paravirtual	562	476	412	448	420
500x500 SAS	551	475	493	502	511
1000x1000 Paravirtual	1619	1724	1807	1755	1740
1000x1000 SAS	2016	2039	1969	2000	1927
2000x2000 Paravirtual	17565	18951	18968	19526	18139
2000x2000 SAS	19889	20318	20329	20354	19233
3000x3000 Paravirtual	46723	40765	44072	42437	42093
3000x3000 SAS	49374	44383	45818	43457	43526

Table 1. SQL table row sample size (rows) vs CPU/NUMA configuration (columns), measurement in milliseconds

Here are our corresponding charts, per sample size:

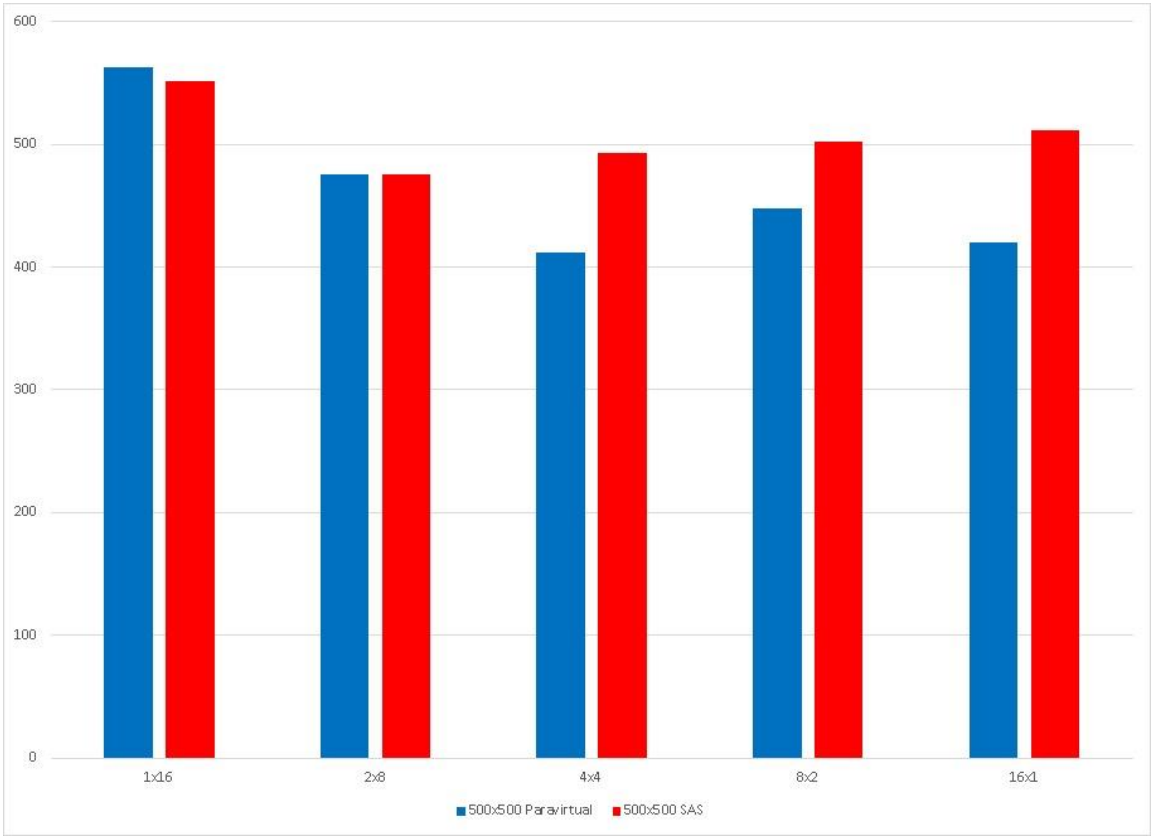

Fig. 1. Performance for 500x500 sample size in milliseconds (lower number = better performance)

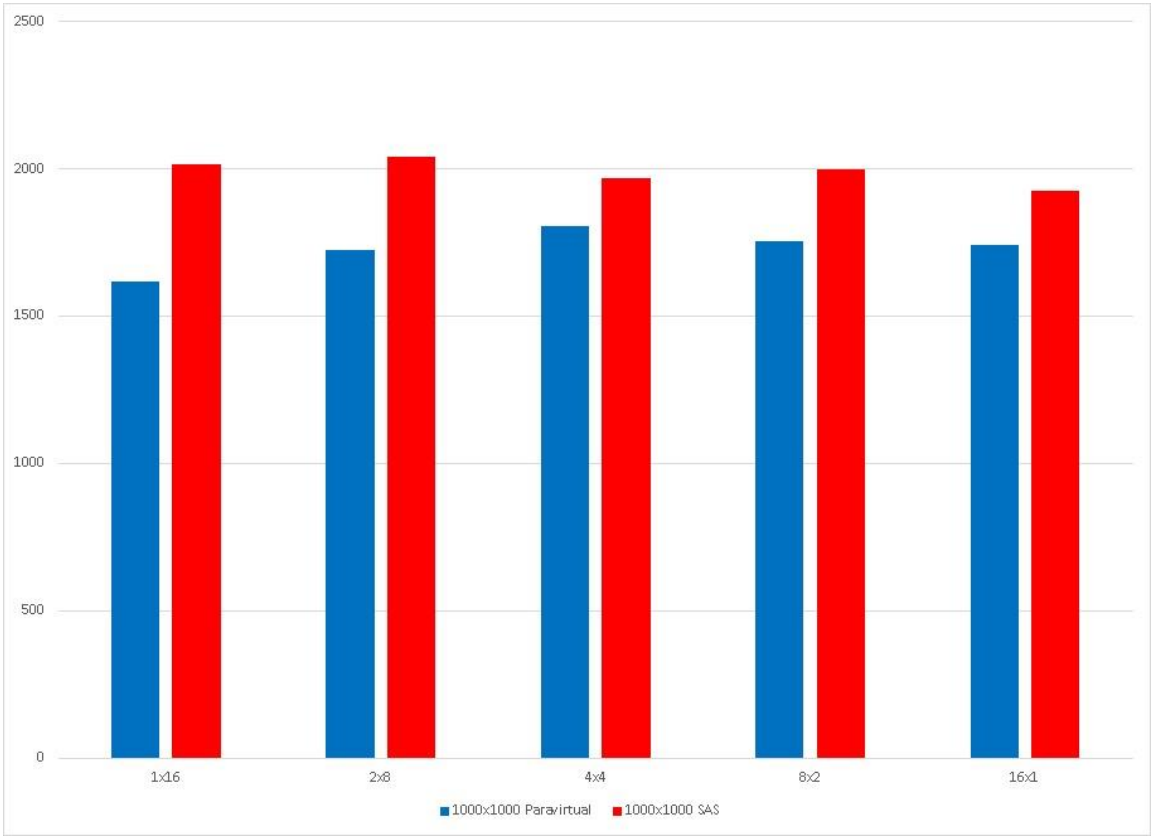

Fig. 2. Performance for 1000x1000 sample size in milliseconds (lower number = better performance)

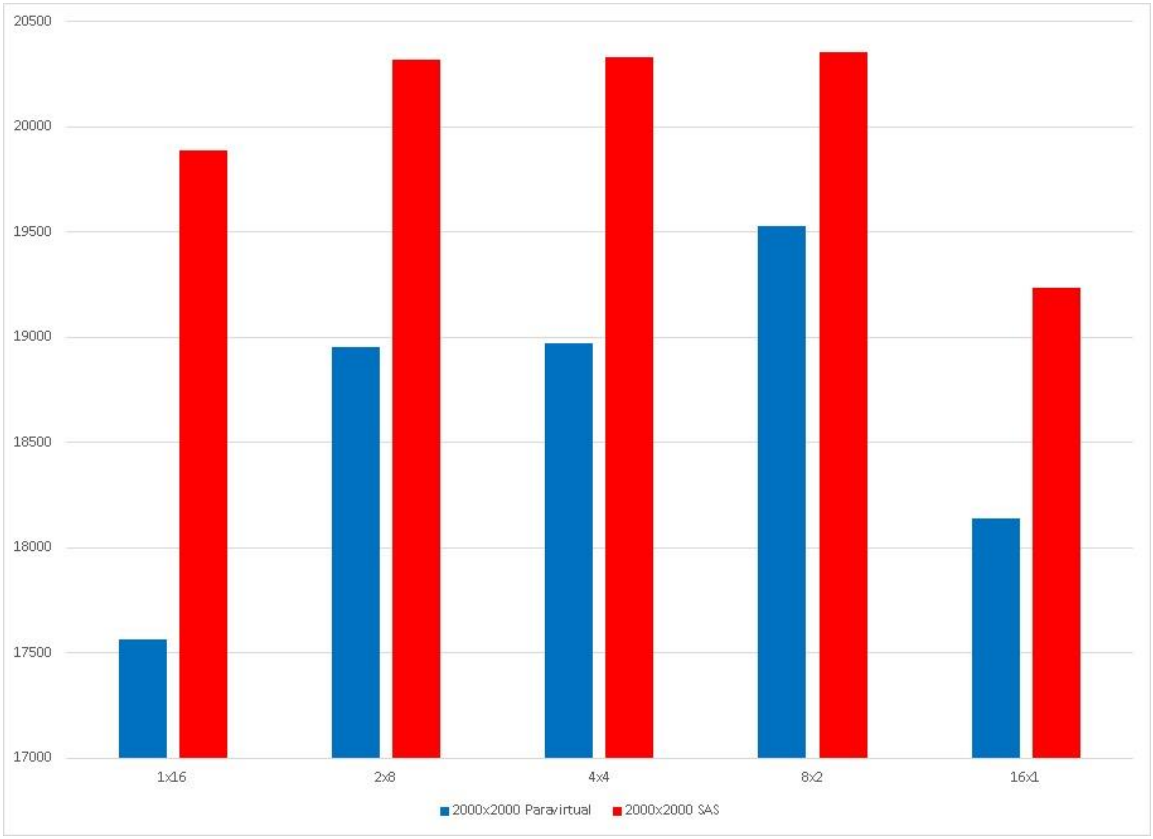

Fig. 3. Performance for 2000x2000 sample size in milliseconds (lower number = better performance)

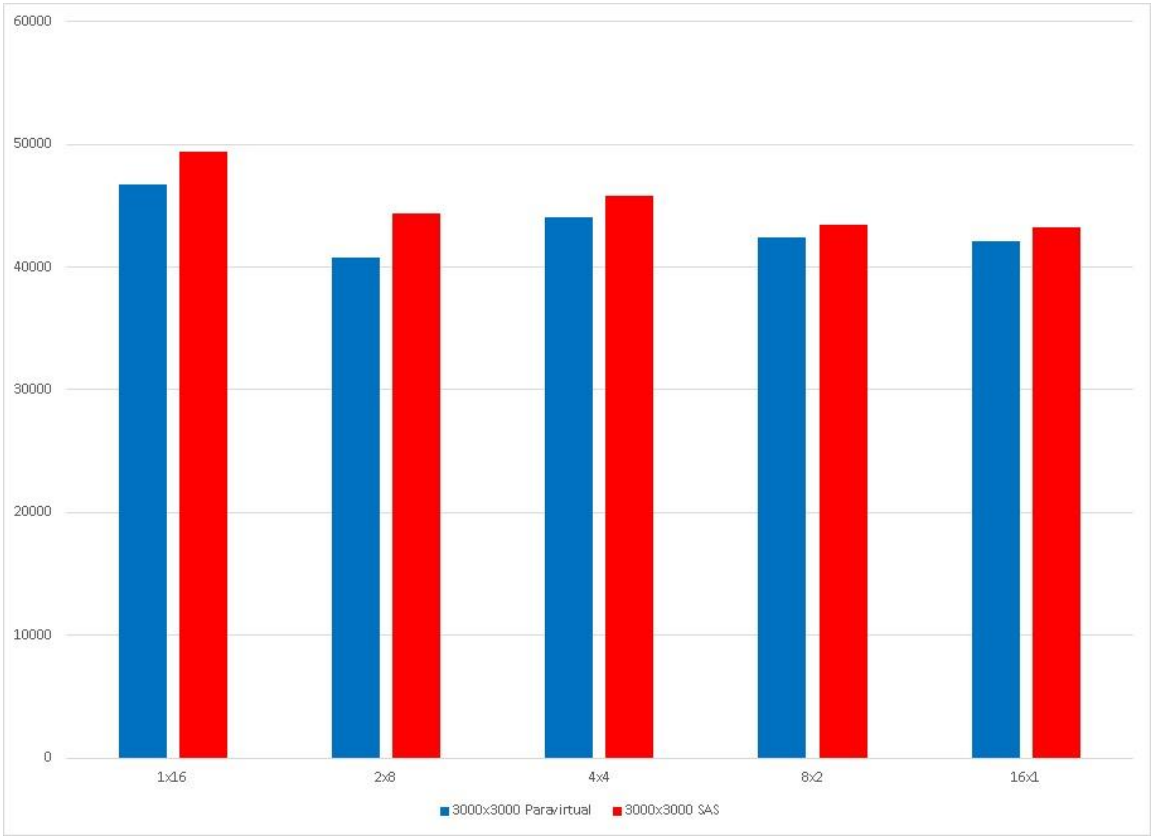

Fig. 4. Performance for 3000x3000 sample size in milliseconds (lower number = better performance)

4. Conclusions

There are definitely more than a few interesting conclusions to me made from our measurements. Overall, performance of the PVSCSI virtual storage adapter isn't all that much better than LSI Logic SAS virtual storage controller. We can actually expand on that, and say that there are situations in which LSI Logic SAS virtual storage controller is faster, especially with smaller sample sizes. However, what VMware Paravirtual storage controller does offer is a significant reduction in CPU overhead on the host level. We measured more than 50% less CPU being used from the host level while using esxtop command and VMware vCenter GUI graphs. At the same time, most of our results point to the fact that VMware Paravirtual storage controller is faster, especially with larger sample sizes. Sometimes, this difference is small - just a couple of percentage points - but there are situations in which it's quite a bit faster. There are results that are up to 12% percent faster while using the same sample size.

On the other hand, there are situations in which we found that VMware Paravirtual storage controller needs a bit of time to "spool up". It doesn't look like it likes the beginning of the I/O process, and it also doesn't look like it likes performing operations on smaller chunks of data, which we might treat as an example of more "randomized" approach. This warrants further study, to see if this type of controller really prefers heavily serialized data and how much does its performance drop if it's doing I/O on randomized, small blocks of data. We suspected that P-state or C-state configuration on our server might be doing something to our test, so we also re-checked our BIOS settings because best practices indicate that it's recommended to use vSphere Power Management and not use any vendor specific BIOS power management profiles [7].

Let's consider for a second a scenario in which we had an environment that uses LSI Logic SAS virtual storage controller exclusively. We might consider a conversion process of our controllers to paravirtual, in search for ultimate performance. There's was realistic chance that doing so might break our virtual machine, and we did encounter a couple of blue screens. Having in mind that this requires a lot of administrative work while adding a significant risk of breaking our machine and offering mostly small amount of performance increase, we hesitate to recommend such a scenario. This kind of scenario is realistic only for environments that are CPU-limited or close to being CPU-limited. Then it might be worth the risk, as this scenario does create a tangible decrease in overhead.

Furthermore, it's very interesting how these virtual storage controllers perform with relation to NUMA configuration, as it brings a significant impact on performance. At times, performance seems completely counter-intuitive, as there are situations in which a properly set-up NUMA configuration brings *lower* performance then a mis-configured NUMA configuration. We checked all of our NUMA-related settings, having in mind that additional NUMA nodes and cores increase OS scheduling and memory allocation complexity [8]. But, having in mind that there are many other variables involved (queue depths, virtual storage controller multi-threaded-ness, etc.), relationship between NUMA settings and virtual storage controller type warrants further research in a future study.

5. References

[1] VMware (2019). Architecting Microsoft SQL Server on VMware vSphere

[2] VMware (2015). Performance Best Practices for VMware vSphere 6.0, VMware

[3] Dakic, V. (2019). Influence of NUMA and memory locality on Microsoft SQL Server 2019 performance, Proceedings of DAAAM., at Zadar, Croatia, ISSN 1726-9679, in press

[4] Randic, M.; Jednakovic, H.& Blaskovic, B. (2010.) Dynamic thread assignment in a tandem of threadpools inspired by the adaptetion mechanism in honeybee foraging, Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Vienna, Austria, ISBN 978-3-901509-73-5.

[5] Saravanan,S; Venkatachalam, V; Malligai, S.T. (2015.), Optimization of SLA Violation In Cloud Computing Using Artificial Bee Colony, International Journal of Advances in Engineering, 2015, ISSN: 2394-9260

[6] Elder, K.; Kusek, Cristopher; Sarkar, P. (2017). vSphere High Performance Cookbook, Packt Publishing, ISBN: 978-1-78646-462-0

[7] Dennerman, F.; Hagoort, N. (2017). VMware vSphere 6.5 Host Resources Deep Dive, Frank Dennerman and Niels Hagoort, ISBN 9781540873064

[8] Hollowell, C.; Caramarcu, C.; Strecker-Kellogg W.; Wong, A. & Zaytsev, A. (2015). The Effect of NUMA Tunings on CPU Performance, Proceedings of 21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015), IOP Publishing, doi:10.1088/1742-6596/664/9/092010